

PUB-NO: EP000019515A2

DOCUMENT-IDENTIFIER: EP 19515 A2

TITLE: Electronic postage meter having improved security and fault tolerance features.

PUBN-DATE: November 26, 1980

INVENTOR-INFORMATION:

NAME	COUNTRY
QUATSE, JESSE TORRANCE	N/A
DODGE, DONALD EVERETT JR	N/A
DOVE, RICHARD KINGSLEY	N/A

ASSIGNEE-INFORMATION:

NAME	COUNTRY
FRIDEN CORP	US

APPL-NO: EP80400603

APPL-DATE: May 5, 1980

PRIORITY-DATA: US03757879A (May 9, 1979)

INT-CL (IPC): G07B017/04, G06F001/00 , G11C011/00 , G06F011/00

EUR-CL (EPC): G07B017/00 ; G06F011/14

US-CL-CURRENT: 235/101, 705/415 , 705/FOR.105

ABSTRACT:

CHG DATE=19990617 STATUS=O> A microcomputerized postage meter (5) that provides high degrees of security and fault tolerance. The meter maintains data security under low power conditions by the use of functionally nonvolatile memory units. Register and other data which must survive normal and abnormal losses of power to the meter electronics are stored in dual redundant battery augmented memories (hereinafter designated BAMs). Upon detecting an error condition, the microcomputer writes an appropriate fault code to the BAMs. A mechanism for disabling the meter includes dual redundant flip-flops which are set to a "faulted" state upon detection by the microcomputer of a failure condition. These flip-flops are powered by the BAM batteries. They cannot be reset except by physical access to the meter interior, which access is only available to authorized personnel at the factory. The fault flip-flops are also set when the microcomputer fails to properly execute its own operating program. Once the meter has been set to a "faulted" state, the fault flip-flops hold two signals, MPCLR and SYSCLR, true. The BAM contents may still be read out independently of the microcomputer which is prevented from accessing the BAMs. This is accomplished by allowing power necessary to read the BAMs to be supplied to the BAMs without supplying power to the microcomputer. Moreover, even if the microcomputer is powered, MPCLR prevents it from executing instructions and SYSCLR isolates it. The various register

values and the fault code then allow a reconstruction of the proper register values.

EUROPEAN PATENT APPLICATION

Application number: 80409603.9

Int. Cl.³: **G 07 B 17/04**
G 06 F 1/00, G 11 C 11/00
G 06 F 11/00

Date of filing: 05.05.80

Priority: 09.05.79 US 37578

Date of publication of application:
26.11.80 Bulletin 80/24

Designated Contracting States:
DE FR GB

Applicant: FRIDEN MAILING EQUIPMENT
CORPORATION
31285 San Clemente
Hayward California 94544(US)

Inventor: Quatse, Jesse Torrance
37, Willow Lane
Sausalito California 94965(US)

Inventor: Dodge, Donald Everett, Jr.
65 Buena Vista Terrace
San Francisco California 94117(US)

Inventor: Dove, Richard Kingsley
180 Maxwellton Road
Piedmont California 94618(US)

Representative: Weinmiller, Jürgen et al,
Zeppelinstrasse 63
D-8000 München 80(DE)

Electronic postage meter having improved security and fault tolerance features.

A microcomputerized postage meter (5) that provides high degrees of security and fault tolerance. The meter maintains data security under low power conditions by the use of functionally nonvolatile memory units. Register and other data which must survive normal and abnormal losses of power to the meter electronics are stored in dual redundant battery augmented memories (hereinafter designated BAMs). Upon detecting an error condition, the microcomputer writes an appropriate fault code to the BAMs. A mechanism for disabling the meter includes dual redundant flip-flops which are set to a "faulted" state upon detection by the microcomputer of a failure condition. These flip-flops are powered by the BAM batteries. They cannot be reset except by physical access to the meter interior, which access is only available to authorized personnel at the factory. The fault flip-flops are also set when the microcomputer fails to properly execute its own operating program. Once the meter has been set to a "faulted" state, the fault flip-flops hold two signals, MPCLR and SYSCLR, true. The BAM contents may still be read out independently of the microcomputer which is prevented from accessing the BAMs. This is accomplished by allowing power necessary to read the BAMs to be supplied to the BAMs without supplying power to the microcomputer. Moreover, even if the microcomputer is powered, MPCLR prevents it from executing instructions and SYSCLR isolates it. The various register values and the fault code then allow a reconstruction of the proper register values.

EP 0 019 515 A2

./...

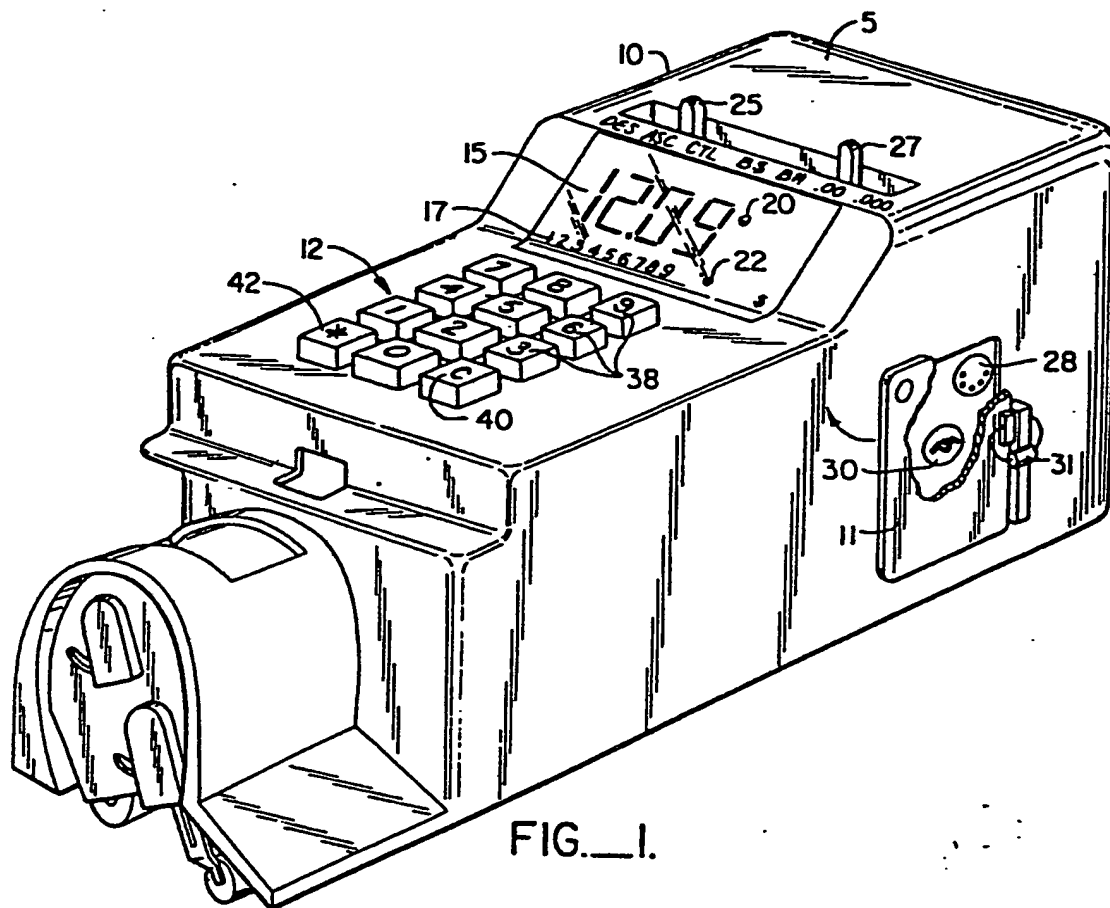


FIG. 1.

6969-3

ELECTRONIC POSTAGE METER HAVING IMPROVED
SECURITY AND FAULT TOLERANCE FEATURES

FIELD OF THE INVENTION

1 The present invention relates to a micro-computerized postage meter.

BACKGROUND OF THE INVENTION

2 Postage meters (hereinafter sometimes designated simply as "meters") are well-known devices for imprinting
3 postage impressions of desired value either on a gummed tape or directly on an article to be mailed, thereby obviating the need to use postage stamps. Due to their convenience and flexibility, meters have found widespread use in commerce.

4 A postage meter normally includes a postage selection mechanism, a postage printing mechanism, and a plurality of internal registers for maintaining accounting information. The internal registers most commonly contain numerical values representative of the total postage paid for (control total), the total postage printed (ascending balance or ascending register), and the total postage remaining (descending balance or descending register). The information contained in the internal registers is redundant, since the ascending balance and descending balance normally sum to the control total.


5 Prior to using the meter a user must buy from a postal service employee a fixed amount of postage. (In this connection, the term "postal service" may refer to either a public or private mail carrying entity.) The postal service employee alters the contents of the internal registers to
6 reflect the amount of postage paid by increasing the control

total and the descending balance by this amount. To use the meter, the user first selects a value of postage to be printed, and then activates the printing mechanism. The meter may be used until the descending balance reaches a
5 predetermined minimum (e.g., until the postage paid for has been exhausted or has reached a minimum threshold value).

It can immediately be seen that postage meters are subject to stringent security requirements to ensure that all postage actually printed has been paid for. Thus, the level
10 of security can be measured by the difficulty of activating the meter's printing mechanism without correspondingly updating the accounting registers within the meter, and also by the difficulty of altering or losing the meter register values, whether intentionally, inadvertently, or accidental-
15 ly. To this end, the print mechanism and the accounting registers are located within a secure housing, and access thereto is restricted to postal service employees.

Postage meters have traditionally been essentially mechanical devices whose mechanical design is relatively
20 complicated due to the need to correlate operation of the postage selection mechanism, the postage printing mechanism, and the registers. In particular, the print mechanism must print a postage value corresponding to the value set by the user, and the appropriate internal registers must be changed
25 by this amount. Moreover, the meter must be interlocked to disable the print mechanism when the descending balance reaches the predetermined minimum level, and to prevent more than a single printing impression from being made during a cycle of the printing mechanism. Mechanisms capable of
30 performing these functions, of necessity, contain a large number of mechanical parts, and therefore require considerable periodic maintenance. While several decades of experience have resulted in the design and implementation of acceptably reliable mechanical postage meters, such devices
35 have still tended to be expensive, heavy, bulky, and slow.

Recent advances in the electronic arts have suggested the desirability of replacing many of the mechanical components in a postage meter with electronic components.




Thus, it is known in the prior art to provide a first-generation electronic postage meter employing discrete logic components. Such a meter is shown in U.S. Patent No. 3,938,095 to Check, Jr. et al.

5 By their nature, electronic postage meters rely heavily on continuous electric power during operation. However, frequent power loss of either a momentary or prolonged nature is to be expected. While power loss is not a particularly significant event for mechanical meters, it
10 poses two distinct threats to the security of electronic postage meters. First, power loss presents a threat to the integrity of the register data which is typically stored in electronic memory units, since most electronic memories are volatile devices (i.e., they require continuous electric
15 power to maintain their contents). This is to be contrasted with mechanical registers which are inherently nonvolatile devices. Second, the various correlation and interlocking functions are performed by electronic logic components, the performance of which can become unpredictable during a low
20 power condition. Since this could lead to improper updating of registers, and the like, there must be provided a reliable mechanism wherein the electronic circuitry inhibits meter functioning when a low power condition is sensed. Moreover, this inhibiting must occur before the power falls to a level
25 at which the electronic circuitry becomes unreliable.

In addition to the security requirements discussed above, a second requirement of postage meters, called "fault tolerance", comes into play when mechanical registers and other functions are replaced by electronic components. Fault
30 tolerance refers to the meter's ability to maintain security in view of individual component failure. A postage meter is likely to be used in a variety of settings that may subject the components to environmental rigors such as mechanical shock, stray electric fields, and wide temperature varia-
35 tions, any of which may cause an electronic, mechanical, or electro-mechanical component to fail.

It is apparent that the large number of components in the first-generation electronic postage meters employing



discrete logic elements (i.e., transistors, diodes, etc.) tends to render such meters insufficiently reliable for postal service approval. A further difficulty with electronic postage meters employing discrete logic components is that the features and capabilities of the meter cannot be altered easily once the meter is constructed. Thus, like their mechanical counterparts, such meters cannot readily be adapted to new applications.

In recognition of the above problems, first-generation electronic postage meters employing discrete logic have given way to second-generation electronic postage meters employing large scale integration microcomputer architecture. An example of such a second-generation stand-alone postage meter is disclosed in U.S. Patent No. 3,978,457 to Check, Jr. et al., employing a microcomputer system which monitors the printing and other functions of the meter, and which supervises and maintains the required accounting information. A microcomputerized postage meter contains a smaller number of components than its discrete component counterpart, and is therefore likely to have improved fault tolerance characteristics. The fault tolerance can be further enhanced by the capability possessed by such a meter of verifying its own functions. Nevertheless, fault tolerance remains a potentially vexing problem because the very components that are used to check for failure are themselves subject to failure.

In spite of the numerous potential advantages of electronic postage meters over their mechanical predecessors, and further in spite of the expanded capability of microcomputerized postage meters, efforts to design an electronic postage meter having sufficiently high levels of security and fault tolerance to obtain postal service approval have been generally unsuccessful to date.

SUMMARY OF THE INVENTION

The present invention is a microcomputerized postage meter that provides sufficiently high degrees of security and fault tolerance to make the meter suitable for postal service approval.

Broadly, the postage meter of the present invention includes a microcomputer system, a keyboard or other suitable input means for entering data and instructions into the microcomputer, a display unit for allowing the operator to
5 . determine the status of particular registers, and a postage printing mechanism. The printing mechanism and the electronic components are located within a secure housing, and unauthorized access is impossible without physical destruction of parts of the housing. The housing mounts on a conventional meter base which performs the various paper
10 handling functions and contains the power supplies.

The meter maintains data security under low power conditions by the use of functionally nonvolatile memory units. Register and other data which must survive normal and
15 abnormal losses of power to the meter electronics are stored in dual redundant battery augmented memories (hereinafter designated BAMS). The BAMS are also used to store historical information for maintenance purposes (e.g., total usage of print mechanism), information relating to particular features
20 or options (e.g., fractional cents), batch information (number of pieces and dollar amounts), and failure information (discussed more fully below). The BAMS are mutually independent, and each BAM includes a low power CMOS memory and a long life (5 years or more) battery. While each BAM
25 can be read independently of the microcomputer, writing of new data into each BAM can only occur under microcomputer program control.

The meter maintains the integrity of its functioning under low power conditions by generating and responding
30 to two timed signals when a low power condition is sensed. A first signal, designated "SYSCLR" (system clear), has the effect of inhibiting all meter functions that could have an effect on the printing of postage or the register values in the BAMS. A second signal, designated "MPCLR" (micro-
35 processor clear), inhibits execution by the microcomputer. The generation of these two signals in a particular time-ordered manner prevents spurious operation during power up and power down periods.

When the supply voltage supplied to the meter falls below a predetermined threshold deemed necessary to ensure continued reliable functioning of the electronic components, a portion of the circuitry initiates a graceful power down sequence. After a sufficient time to allow completion of any ongoing BAM register updates (about 20 milliseconds), a SYSCLR signal is generated, i.e., SYSCLR goes true, inhibiting writing to the BAMs, and disabling the print mechanism. Then, after a delay (about 1 millisecond), an MPCLR signal is generated, i.e., MPCLR goes true, and the microcomputer is disabled. A capacitor within the meter retains sufficient charge at a voltage above that required by the electronic components to ensure that the circuitry operates reliably during the power down sequence.

An analogous sequence occurs when power is first applied to the meter circuitry during a power up cycle. When the supply voltage to the meter electronics is non-zero but still below the predetermined threshold level, MPCLR and SYSCLR go true. After a sufficient interval after the time that the voltage has risen to the predetermined threshold value to ensure reliable microcomputer operation (about 50 milliseconds for the particular microcomputer used), MPCLR goes false, which allows the microcomputer to start executing an initialization routine. Then after a delay (about 2 milliseconds) SYSCLR goes false to permit normal meter functioning.

The basic operating philosophy of the postage meter is to maintain security under normal operating conditions by having the microcomputer supervise and verify the various meter functions.

Printing is strictly supervised by the microcomputer. The meter employs a print mechanism comprising a print head which includes a plurality of print wheels, and a plurality of stepper motors corresponding to the plurality of print wheels, each motor controlling the positioning (indexing) of an associated one of the plurality of print wheels. Each stepper motor has verification means for generating a digital code, preferably binary coded decimal (BCD) code,

corresponding to the print wheel position. Locking means, such as a solenoid, under exclusive control of the microcomputer maintains the print head locked up in the home position except when printing is to be carried out in accordance with instructions from the microcomputer. Before freeing the print head for printing, the microcomputer verifies that the print wheels are in the positions corresponding to the desired amount of postage to be printed.

Changes to the register values in the BAMS can only occur under control of the microcomputer. In regular use, the meter is in a condition designated "mail room" mode wherein the only instructions available to the operator for changing the register values stored in the BAMS are the instructions for initiating a postage printing cycle. The effect of the print cycle is to decrement the descending register and increment the ascending register by the amount of postage printed. Register updating occurs only upon the receipt of a signal from the base that a clutch in the motor that drives the print head had been pulled or an internal signal that the print head has left its home position. As discussed above, the print head is freed only after the print wheel positions have been verified. Only authorized personnel can cause the microcomputer to execute a series of instructions to increase the control total value and descending register value by an amount corresponding to an amount of postage that is being added to the meter. Such instructions can only be executed when a mode-changing switch is actuated to switch the meter to a condition designated "post office" mode. In order to actuate the mode-changing switch, a seal must be broken and the switch unlocked with a special key available only to authorized personnel. The meter is not mounted on a base in the "post office" mode since no printing is to occur but rather, the meter is provided with external power to operate the electronic control circuitry.

All updating of register values, whether at the initiation of the print cycle or at the time of purchase of additional postage, occurs according to a checked arithmetic algorithm. The basic arithmetic constraint is that the

ascending and descending registers sum to the control total. The mechanics of the checking and the updating differ somewhat, depending on whether the meter is in the "mail room" or "post office" mode. In both cases, an extra "temporary" copy of the ascending and descending register values and control total is stored in the BAMs, in addition to the redundant "permanent" values. A consistency check is made to ensure that the register values in one BAM agree with the corresponding values in the other BAM. In the "mail room" mode the temporary values are updated as each character is entered on the keyboard; in the "post office" mode updating of the temporary registers occurs only after the entire keyboard entry is complete. At each update of the temporary register values, the values are checked for adherence to the arithmetic constraint. Updating of the permanent registers occurs during the print cycle in the "mail room" mode and in response to a particular keyboard sequence in the "post office" mode. The updated permanent registers are then checked for agreement between BAMs and for adherence to the arithmetic constraint.

It is apparent that the use of redundant BAMs, the maintenance of temporary register values in the BAMs, and the manner of updating the BAM registers prevent the loss of correct register values in the event of many types of malfunction. For example, if the microcomputer were to cease execution while updating the permanent registers in the BAMs in either the "post office" or "mail room" mode, the temporary registers would still provide sufficient redundant information to permit retrieval of the correct register values.

However, once a malfunction has occurred, the meter no longer possesses the redundancy necessary to maintain security in the event of a further malfunction. Accordingly, there is provided a mechanism for disabling the meter once a first malfunction is detected. This results in a meter that possesses a level of fault tolerance wherein security is maintained in spite of a single malfunction. The meter is still susceptible to loss of the correct register values if two independent malfunctions occur in a particular comple-

mentary way (e.g., loss of power to one BAM and memory failure of the other BAM; simultaneous loss of power to both BAMs; compensating arithmetic errors in both of the BAMs). Such combinations of malfunctions are extremely rare since
5 the individual malfunctions are basically independent occurrences that are themselves rare.

The meter disabling mechanism includes dual redundant flip-flops which are set to a "faulted" state upon detection by the microcomputer of a failure condition. These
10 flip-flops are powered by the BAM batteries. The setting of these fault flip-flops lights an indicator lamp and generates a SYSCLR signal to inhibit all meter functions that could have an effect on the register values in the BAMs. In addition, the setting of the fault flip-flops generates an MPCLR
15 signal to cause the microcomputer to cease execution. Setting of the fault flip-flops is inhibited by the presence of a SYSCLR signal, so that the fault flip-flops cannot be set during power up and power down cycling. The fault flip-flops, once set, hold SYSCLR and MPCLR true whenever power is
20 supplied to the meter. They cannot be reset except by physical access to the meter interior, which access is only available to authorized personnel at the factory.

The fault flip-flops are also set when the microcomputer fails to properly execute its own operating program.
25 Under proper conditions, the microcomputer periodically generates a signal which triggers two independent monostable multivibrator circuits, preferably retriggerable one-shots, to maintain a particular logic level on the one-shot output. The microcomputer responds to the detection of a fault condition by not triggering the one-shots. When there appears on
30 the one-shot output a logic level that corresponds to the failure of the microcomputer to generate this signal, the fault flip-flops are set. In addition to programmed failures to trigger the one-shots, any circumstance (such as microcomputer failure) that prevents the microcomputer from triggering the one-shots has the effect of causing the meter to
35 fault. Thus, the mechanism for setting the fault flip-flops exploits the microcomputer's ability to diagnose failures but may also be activated should the microcomputer itself fail.

Adherence of the register values to the arithmetic constraint, and consistency between the BAMs, as described above, are considered absolute prerequisites to continued meter operation. Thus if an arithmetic check yields inconsistent results, or if corresponding registers in the two BAMs disagree, the meter faults after writing an appropriate failure code to the BAMs. All writing to the BAMs is verified, and an error causes the meter to fault, preceded by the writing of an appropriate failure code.

During the printing operation, the microcomputer and associated logic elements test for various indications of malfunction of the print value setting and printing mechanisms. In particular, if during a printing operation the print head leaves its home position before the print wheels are positioned or the print head fails to leave its home position within a predetermined maximum time period (100 milliseconds) of the receipt of a clutch signal, the meter faults. In such cases, a numeric code representative of the particular problem detected is written to the BAMs prior to deactivation of the meter.

Other types of malfunction do not pose an immediate threat to security, and tend to be self-curing or one-time occurrences. These situations are handled by turning off the displays and stepper motors, locking the print solenoid, and causing the microcomputer to execute a trivial wait loop while still continuing to fire the one-shots. This condition is designated a "soft fault". The typical cause of a soft fault is some sort of mistake which necessitates suspending operation to prevent a second mistake which could cause an error. The mistake may arise from operator error, base malfunction, dirt, or noise. An example of soft fault is the apparent failure of a print wheel to move from one position to an adjacent position within a predetermined maximum time period (12.8 milliseconds). In many instances, the cause might be due to imperfect operation of the verification means, such as poor electrical contact due to dirt. In such a case, reinitializing the meter which causes the motors to be stepped through their entire range would cure this

problem. Another example of a soft fault is the detection of a clutch signal when it is not expected. This could be caused by a malfunction in the base or it could result from improper paper handling by the operator. Again, reinitializing the meter would cause this problem to go away. A further example of a soft fault is the inability to verify a BAM address. Since the BAM address lines communicate outside the meter housing, they are susceptible to electrical noise which could cause a bad address verification. Again, this problem is likely to be intermittent in nature and does not pose a threat to security so long as no attempt is made to write to the BAMs.

Yet a further potential threat to the meter's security is present if the mode-changing switch malfunctions, giving a spurious indication to the microcomputer that the meter is in the "post office" mode since this could allow an unauthorized (unpaid for) increase of the control total and descending registers. To prevent such an occurrence, the microcomputer checks the mode-changing switch during initialization and at the beginning of a print cycle. If the meter is found to be in the "post office" mode while apparently mounted on a base at either of these times, a soft fault condition is considered to have occurred. One possible cause of such a condition is if a postal service employee actuates the mode changing switch while the meter is mounted on a base. So long as operation is suspended, no threat to security exists, and the soft fault condition is cured by having the postal service employee reinitialize the meter under external power (meter off base) or on the base but in the "mail room" mode.

During initialization, the microcomputer checks to make sure that no nonzero failure codes have been written to the BAMs. Once an error code has been written to the BAMs and the meter has faulted, it should be impossible to initialize the meter since the fault flip-flops hold MPCLR true when power is supplied. Thus, detection of a nonzero failure code is a possible indication of a BAM failure or an inability to read a good zero from the BAMs, and causes the meter

to fault. Due to the nature of the error, no attempt is made to write a failure code.

Once the meter has been set to a "faulted" state, the BAM contents may still be read out independently of the microcomputer. During this time, the microcomputer is prevented from accessing the BAMs. This is accomplished by allowing power necessary to read the BAMs to be supplied to the BAMs without supplying power to the microcomputer. Moreover, even if the microcomputer is powered, MPCLR prevents it from executing instructions and SYSCLR isolates it. The various register values and the fault code then allow a reconstruction of the proper register values.

For a further understanding of the nature and advantages of the invention, reference should be had to the ensuing detailed description taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a perspective view of the exterior of a microcomputerized postage meter according to the present invention;

Fig. 2 is a perspective view of part of the print mechanism located within the housing of the meter;

Fig. 3 is a schematic block diagram of the electronic meter system;

Fig. 4 is a system flow chart illustrating the general operation of the invention;

Fig. 5 is a circuit diagram of the external power levels that are communicated from the base to the meter;

Fig. 6 is a circuit diagram of signals communicated from the base to the meter;

Fig. 7 is a circuit schematic of circuitry for generating certain internal signals within the meter;

Fig. 8 is a circuit schematic of the power surveillance and system reset circuitry of the present invention;

Fig. 9 is a circuit schematic of the circuitry that inhibits meter functioning upon detection of a fault condition;

Fig. 10 is a timing diagram illustrating the voltage levels responded to by the power surveillance circuitry of Fig. 8, and the signals generated by the system reset circuit of Fig. 8;

5 Figs 11a, 11b, and 11c, taken together form a circuit schematic illustrating the microcomputer and BAM control circuitry;

Fig. 12a and 12b, taken together, form a circuit schematic illustrating the microcomputer circuitry for
.0 refreshing the displays and reading the switches;

Fig. 13 is a circuit schematic illustrating circuitry for controlling the print wheel motors and reading the verification contacts;

Fig. 14 is a schematic illustrating a suitable
.5 memory allocation of registers required by the operating program of the microcomputer;

Fig. 15 is a schematic illustrating a suitable bit allocation for the switch registers;

Fig. 16 is a schematic illustrating a suitable bit
20 allocation for the status and print routine registers;

Fig. 17 is a flow chart of the foreground routine;

Fig. 18 is a schematic illustrating the organization of the BAMs and the memory allocation of register values stored therein;

Fig. 19 illustrates in tabular form the temporary
25 register contents after various operations;

Figs. 20a, 20b, and 20c, taken together, form a flow chart of the print task routine;

Figs. 21a and 21b, taken together, form a flow
30 chart of the keyboard task routine.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Turning now to the drawings, Fig. 1 is a perspective view of the exterior of a postage meter 5 according to the present invention. Meter 5 includes an exterior housing
35 10 for containing a print mechanism and electronic control system described in detail below. In operation, meter 5 is installed on a mailing machine base, not shown, which contains power supplies for the meter, performs paper handling

functions (envelopes or tapes), communicates synchronizing signals to the electronic control system, and provides mechanical power for activating (but not setting) the print mechanism. Since the mailing machine base comprises a conventional ancillary device, a detailed description of the elements thereof is omitted to avoid prolixity.

The exterior components of meter 5 generally include a control panel and a door behind which is a postal service employee's panel. The control panel includes a keyboard 12, displays 15 and 17, indicator lights 20 and 22, and selector switches 25 and 27. The postal service employee's panel includes a remote power connector 28 for powering the meter when it is removed from a base, and a keylock operated mode changing switch 30 which, when actuated, places the meter in a "post office" mode to permit the postal service employee to change the register values to reflect a purchase of additional postage. Access to connector 28 and switch 30 requires the breaking of a seal 31 and subsequent sliding of door 11.

The plurality of keyboard keys, switches 25, 27, and 30, and displays 15 and 17 allow the operator or postal service employee to communicate with the internal electronic control system and to specify the necessary control signals for operation of the electronic control system and the print mechanism. Meter 5 may also operate under control of external devices (e.g. an electronic weighing station) and thus includes suitable connections for communicating electrical signals to the electronic control system within housing 10. Hereinafter, the term "switches", when used generally, will be taken to include the above switches and keys, the base synchronization signals, and signals from external devices.

Keyboard 12 comprises a calculator configured digit field including ten numbered keys 38, and special function keys 40 and 42, designated clear KB and clear batch, respectively. Data entry is effected by manually actuating numbered keys 38 in a sequence corresponding to the desired entry. The most significant digit is entered first, normally with a maximum dollar entry capacity of four digits (99.99 or

9.999 if a fractional cents meter). Attempts to continue entry beyond the four digit maximum are ignored. This entry capacity expands to up to eight digits when the meter is in the "post office" mode, the maximum number of digits being a
5 preset parameter.

Display 15 is preferably a four-digit numeric display having characters sufficiently large to be readily visible in normal operation. A character height of about 0.6 inches is suitable. As data is entered via keys 38, it is
10 displayed on display 15. As values are keyed in, they enter the right-most digit and are shifted left for each valid subsequent keyboard entry. Depression of clear KB key 40 clears the keyboard entry and display 15. Simultaneous depression of clear batch key 42 and clear KB key 40 initial-
15 izes internal batch count and batch dollars registers. Display 17 is preferably a nine-digit numeric display and is used for displaying register values. Since this display is generally used less often than display 15, a smaller character size (e.g., approximately 0.125 inches) than that used
20 for display 15 is suitable. Five position slide switch 25 allows the user to select an internal register to be displayed on display 17. In particular, descending register value, ascending register value, control total value, batch dollar amount, and batch count may be displayed. Two-
25 position selector switch 27 has effect only if meter 5 has been pre-programmed as a "fractional cents" meter. In effect, switch 27 allows the operator to override the fractional cents aspect during keyboard entry so that the last digit entered corresponds to cents rather than tenths of a
30 cent.

When lit, indicator lights 20 and 22 serve to warn the operator that a condition has occurred which makes it necessary to at least temporarily suspend operation of the meter. Indicator light 20, designated "Service", is lit
35 when the meter has been internally set to a faulted state. When light 20 is lit, the meter is inoperative, and must be returned to the factory before it can be used again. In addition, no set of keyboard entries or sequence of powering

the meter can restore operation, and access to the housing interior, possible only at the factory, is required to place the meter in operative condition. Indicator light 22, designated "Add \$", is lit when the amount of postage entered would cause the descending register value to become negative upon printing. Depressing "clear entry" key 40 extinguishes light 22 and allows subsequent meter operation at postage values which will not cause the descending register value to become negative.

10 The amount of postage available for printing may be changed by a postal service employee by placing meter 5 in the "post office mode". This is possible only when the postal service employee breaks seal 31 on switch 30, and enables movement of switch 30 to the "post office mode" position by actuation of keylock 32. When this occurs, keys 15 40 and 42 and switch 25 take on different functions. When switch 25 is moved to either of the batch register display positions, nine-digit display 17 functions as a keyboard verification display. Keyboard keys 38 retain their normal function of data entry, but special keys 40 and 42 allow the 20 postal employee to change the values of the accounting registers. In particular, depression and release of clear batch key 42 causes the keyboard entry to be added to the descending register and the control total values. If clear 25 KB key 40 is depressed while clear batch key 42 is also depressed, the keyboard entry is subtracted from the descending register and control total values when "clear batch" key 42 is released.

Fig. 2 is a perspective view of portions of the print mechanism located within housing 10 of meter 5. The 30 print mechanism comprises a print head having a plurality of print wheels, a plurality of stepping motors (hereinafter sometimes referred to as steppers) to independently set the print wheels, and means for moving the entire print head to a printing position without disturbing the print wheel set- 35 tings. Fig. 2 shows the print wheel setting means for a single digit. A print wheel 43 having a plurality of value indicating print surfaces 44 is driven by a stepping motor

45. Additional motors 46, 47, and 48, while not shown in Fig. 2, are shown schematically in Figs. 3 and 13. Stepping motor 45 drives a gear 49 which engages a rack 50. Rack 50 is rigidly connected to a drive yoke 52 which carries a
5 rotatably mounted drive ring 55. Drive ring 55 is slidably mounted on a rotatable bar 57 which has a channel 60 along its length. A rod 62 is disposed within channel 60, being coupled to drive ring 55 at one end and terminating in a rack portion 65 at the other end. Print wheel 43 has a gear 68,
0 and rack 65 drives print wheel 43 through gear 68 and an idler gear 70.

Thus, rotation of gear 49 on the output shaft of motor 45 causes movement of rack 50 along the direction of bar 57. Movement of drive yoke 52 is transmitted through rod
5 62 and idler gear 70 to gear 68 and print wheel 43. Motor 45 is coupled to a position sensor 75, which in the preferred embodiment comprises verification contacts for generating a binary code representative of the position of an internal wiper connected to the motor shaft. Additional position
10 sensors 76, 77 and 78 are associated with motors 46, 47, and 48, respectively, and while not shown in Fig. 2, are shown schematically in Figs. 3 and 13.

It should be understood that a separate subassembly comprising a stepping motor, a motor gear rack, a position
15 sensor, a drive yoke, a drive ring, a rod, a print wheel rack, an idler gear, and a print wheel substantially similar to elements 45, 50, 75, 52, 55, 62, 65, 70, and 43 is provided for each digit that can be printed. Thus, in the preferred embodiment where four digits may be printed, there
30 are four such subassemblies. However, there is only one channeled bar 57, channel 60 being sized to accomodate four rods (including rod 62) in a side by side configuration.

Printing occurs by a rotation of bar 57, the mechanical power being supplied by the base. It should be noted
35 that the drive yokes, while movable along bar 57, are prevented from rotating. Rather, bar 57 rotates independently. The print head (which includes the print wheels) is maintained in a fixed relationship with respect to bar 57 so that

rotation of bar 57 brings the print head from a normally upwardly facing disposition into a downwardly facing position for printing, during which the print wheels are exposed at an opening in the bottom of housing 10. A solenoid 85, shown schematically in Fig. 12 prevents movement of the print head in the absence of a specific print enabling signal.

Fig. 3 is an electrical schematic showing in functional block diagram form the electronic circuitry located within meter 5. Whenever meaningful, reference numerals corresponding to those associated with the physical representation of the same components in Figs. 1 and 2 are used.

The central element in the meter electronic circuitry is a microcomputer 90 having contained program memory, working memory, and processing capabilities. During operation, microcomputer 90 is responsible for performing several functions, including periodic refreshing of the displays, periodic sampling of all the switches, communication with external devices, detection of synchronizing signals from the base that signify the initiation of a print cycle, initiating and verifying print wheel setting, and performing accounting functions. Since microcomputer 90 is required to communicate with a relatively large number of peripheral devices, it operates in conjunction with input/output (hereinafter designated I/O) expansion circuitry. The I/O expansion circuitry includes I/O expanders 92 and 93, demultiplexers 94 and 95, and multiplexers 96 and 97. As will be described in detail below, I/O expander 92 provides signal paths, via demultiplexers 94 and 95 and multiplexer 96, between microcomputer 90 and the displays and switches. The switch status is communicated to microcomputer 90 on an output line 98 of multiplexer 96. In a like fashion, I/O expander 93 allows microcomputer 90 to send signals for driving the stepping motors, while multiplexer 97 allows microcomputer 90 to receive information on a line 99 from the motor position sensors.

In addition to the switches, displays, motors, position sensors and I/O expansion circuitry described above, the circuitry includes dual redundant battery augmented

memories 100 and 102 (BAMs). The BAMs are used to store information which must survive normal and abnormal losses of electrical power to the meter. This includes register data (ascending register value, descending register value, control
5 total value), batch count information, and failure information, all to be discussed in greater detail below.

The circuitry also includes power surveillance circuitry 105 for detecting what is anticipated to be a low power situation, and system reset circuitry 110 that responds
10 to low power conditions by generating timed signals, designated "SYSCLR" (system clear) and "MPCLR" (microprocessor clear), in a particular time ordered manner for a graceful power up or power down sequence. Dual redundant fault flip-flops 111 and 112 are coupled to the system reset circuitry
15 to inhibit meter functioning once the flip-flops are set to a "faulted" condition. Flip-flops 111 and 112 are maintained in a non-faulted state by the outputs of dual redundant retrig-gerable one-shots 113 and 114, respectively. During operation, so long as no failure condition is detected, microcom-
20 puter 90 periodically sends a signal to one-shots 113 and 114 to prevent flip-flops 111 and 112 from being set. Setting of flip-flops 111 and 113 occurs whenever the microcomputer fails to send a signal to one-shots 113 and 114 within a predetermined time period. This occurs either when the
25 microcomputer detects any one of a number of failure indicating conditions, or when the microcomputer itself fails to properly execute its operating program.

Microcomputer 90 communicates with the remaining circuitry primarily via three eight-bit data buses 120, 125,
30 and 130, designated DB, P1, and P2 respectively. Individual lines on buses 120, 125, and 130 are designated DB0-DB7, P10-P17, and P20-P27, respectively. Buses 120 and 125 communicate with external peripheral I/O devices, designated generally by reference numeral 131. I/O devices 131 communi-
35 cate signals which cause microcomputer 90 to carry out the same tasks as it would in response to corresponding switch manipulations. Within meter 5, certain of these data lines serve dual functions. In particular, bus 125 communicates

with BAMS 100 and 102 for supplying an address, and with displays 15 and 17 through a buffer 128 on an eight-bit line 129 for supplying a digit code. Also, lines 132 and 133 of bus 130 serve the dual function of selecting which of BAMS
5 100 or 102 is to be accessed and of selecting which of I/O expanders 92 and 93 is to be accessed via four-bit line 134 of bus 130. Additionally, a four-bit line 135 from I/O expander 92 communicates with the BAMS for supplying data and further communicates with multiplexer 97 for selecting a
10 particular bit from one of motor position indicators 75-78.

Having discussed the general structure of the meter electronic circuitry, the general operation of meter 5 under the control of microcomputer 90 may be described. Broadly, microcomputer 90 executes instructions according to an operating system which includes an initialization routine, a
15 foreground routine, a power loss interrupt routine, a background dispatcher loop, and a plurality of background tasks, including subroutines. The background dispatcher loop is, in effect, the main program, and will sometimes be referred to
20 as such. Fig. 4 is a flow chart illustrating the relationship between the initialization routine, the background dispatcher loop, and the background tasks.

Given the conflicts on buses 120 and 130 it is necessary that data or addresses on these lines that could
25 affect access to the BAMS not be disturbed by an attempt to use one of the lines for its alternate function. Thus, the operating system of microcomputer 90 maintains a sharp division between those portions of the program that access the BAMS (designated background) and those portions of the
30 program according to which microcomputer 90 communicates with the switches, displays, and the like (designated foreground).

The background dispatcher loop and the various background tasks are responsible for the normal meter functions including print supervision and accounting, and rely on
35 information supplied by the foreground routine. The foreground routine is executed in response to a periodic timer interrupt generated by a timer inside microcomputer 90 and has the basic functions of energizing the displays, reading

the switches and base synchronization signals, and sending signals to one-shots 113 and 114 to prevent fault flip-flops 111 and 112 from faulting the meter. The state of the various switches ascertained by the foreground routine is stored
5 in switch registers within the working memory of micro-computer 90, so as to provide information to other parts of the operating system. The basic constraints on the timer interval are imposed by the requirements that the display digits be energized a particular number of times per second
10 and that the keyboard switches be read more often than the minimum duration of a switch closure. As will be seen below, the rate at which the one-shots must be triggered is an easily varied parameter.

Upon receipt of a signal from power surveillance
15 circuitry 105, an interrupt is generated, and the power loss interrupt routine is entered. This routine blanks displays 15 and 17, deenergizes stepping motors 45, 46, 47, and 48, and sets solenoid 85 to the print disabling position. The routine also sets a bit in memory, in analogy to the switch
20 registers to communicate to the background that a power loss interrupt has occurred.

Turning to the background, with reference to Fig. 4, after execution of the initialization routine (block 150), microcomputer 90 sets a bit in its memory to signify that no
25 BAM accesses are in progress, this state being designated "enable foreground" (block 152), and enters the background dispatcher loop indicated generally within dashed rectangle 154. Within this loop, microcomputer 90 sequentially checks the status of the power loss bit and various so-called WAKEUP
30 bits that will have been set by the foreground routine, and jumps to an appropriate background task as required. The first check is whether the power loss bit has been set (block 158). If it has, microcomputer 90 executes a wait loop 160 until the system reset circuitry generates an MPCLR signal to
35 cause microcomputer 90 to cease execution. Assuming no power loss, microcomputer 90 sequentially checks the WAKEUP bits to determine whether a register display is required (logical branch 162), whether the meter is in a print cycle (logical

branch 164), whether an I/O request has been made (logical branch 166), whether the keyboard register has changed indicating a new keyboard entry (logical branch 168), and whether a print task is to be performed (logical branch 170), before
5 returning to the beginning of the background dispatcher loop. During a print cycle, I/O requests and new keyboard entries are ignored, as indicated by branch 171.

Assuming that a register is to be displayed, that an I/O request is present, that a new keyboard entry has been
10 made, or that a print task is to be performed, the program jumps to the appropriate background task indicated by blocks 172, 176, 178, and 180 respectively. The particular features of the background task routines will be described in detail below. Since execution of each of these background tasks
15 involves access to the BAMs at some point, the foreground is immediately disabled so that data on the lines to the BAMs will not be overwritten during the execution of the foreground routine which occurs in response to a timer signal not under the control of the background dispatcher loop. After
20 completion of the appropriate tasks, each of the background routines returns to the background wait loop after enabling the foreground (block 152). Under certain circumstances, the print task routine does not return directly to the main background loop, but first jumps to a portion of the keyboard
25 task return before returning (blocks 182 and 183).

A detailed description of the foreground routine is deferred until after the structure of the meter electronic circuitry has been set forth in greater detail. Nevertheless, the following general description is set forth in order
30 to make clear the overall operation of the meter operating system. The foreground routine does not accomplish all its tasks on a single entry. Rather, a total of 27 timer interrupts are required for a complete cycle. Thus a 400 microsecond timer interval results in a cycle time of about 11
35 milliseconds, which correlates with the need for the digits of display 17 to be energized approximately 80 times per second. Also, the 11 millisecond interval is considerably shorter than the 50 millisecond minimum duration of a pulse resulting from the depression of a keyboard key.

The digits on display 15, requiring greater brightness than those on display 17 must be energized more often. In particular, display 15 is energized four times for every time that on display 17 is energized. As will be discussed
5 in greater detail with reference to Figs. 12a and 12b, sending a signal to energize a particular digit on display 15 or 17 allows a particular position of register display selector switch 25 or a particular keyboard switch 38 to be read. The state of the remaining switches must be separately deter-
10 mined. Since switch noise having a duration of approximately 2 milliseconds is inevitably present during the initial actuation of a switch, the foreground routine debounces the switches and does not consider a switch state to have changed unless it remains changed on two successive readings (i.e.,
15 for more than 11 milliseconds).

Upon entering the foreground routine in response to an interrupt from the 400 microsecond timer, microcomputer 90 checks the foreground bit to determine whether the foreground is enabled or whether a background task was in progress. If
20 the foreground is enabled, the routine does the appropriate display energization and/or switch state read according to a sequence to be described below. Assuming the foreground is enabled, the routine then checks whether the interrupt is a particular one in the sequence that requires a signal to be
25 sent to the fault one-shots and only triggers the one-shots at that particular point in the sequence. If the foreground was not enabled, the routine triggers the fault one-shots regardless of the interrupt's position in the sequence control and then passes to the background routine.

30 The discussion immediately following is with reference to Figs. 5 - 13 which are circuit schematics illustrating in greater detail the electronic meter system set forth in the block diagram of Fig. 3. Accordingly, in the discussion that follows, reference should, at appropriate times,
35 also be made to Fig. 3. The circuit elements are, wherever practical, solid state integrated circuit components. Suitable components are set forth in the following table:

	<u>Function</u>	<u>Type</u>	<u>Manufacturer</u>
	Microcomputer	8049	Intel
	BAM	P5101L-1	Intel
	I/O Expander	P8243	Intel
5	Comparator	LM2903P	National Semiconductor
	Buffer	ULN2003AN	Texas Instruments
	One-shot	MC14538BP	Motorola
	NAND gate	MC14011P	Motorola
	NOR gate	MC14001	Motorola
10	OR gate	MC14071	Motorola
	Multiplexer	74C150	National Semiconductor
	Demultiplexer	74145	Texas Instruments

It should be noted that the preferred type of microcomputer (Intel microcontroller 8049) includes 2048 bytes of read only memory (hereinafter designated ROM) which is programmed with the desired operating program by the manufacturer. During the period that the invention was undergoing development, a type 8035 microcontroller having no ROM of its own was used in connection with an Intel 8755A memory chip which provides 2048 bytes of UV programmable ROM, to facilitate changes to the operating program.

Figs. 5 and 6 are circuit diagrams illustrating external power levels and signals that are communicated from the base to the postage meter. Communication is preferably established by a multiple conductor cable having suitable connectors at its ends to mate with corresponding sockets on the base and on the meter.

Turning first to Fig. 5, various power levels are shown. An external source of at least +9.6 volts, designated VLOGIC is coupled to a line 200 in meter 5. The voltage on line 200 charges a large (e.g. 10,000 microfarad) capacitor 201. Line 200 communicates to an input of a 5-volt regulator 202 to produce a regulated +5 volt power level, designated VCC, on a line 205. VCC is used for powering various integrated circuit components within meter 5 that require a regulated +5 volt source.


BAMs 100 and 102 are powered by separate voltage sources, designated VBAM1 and VBAM2. These voltages are

produced on respective lines 210 and 212, within meter 5. A related +5 volt level, designated VBAM0, is generated on a line 215. A first pair of 2.8 volt batteries 220 is coupled to line 210 via three diodes 225 in series. Similarly, a
5 second pair of 2.8 volt batteries 230 is coupled to line 212 through three diodes 235 series. VLOGIC is coupled to a line 240 within meter 5, and is regulated to +5.7 volt by a 5 volt regulator 242 and a diode 245 to ground. The regulated output voltage appears at a circuit point 250 and is communi-
10 cated to line 215 through a diode 251. Circuit point 250 is coupled to line 210 through a diode 252 and to line 212 through a diode 255.

Two levels, designated VEXT1 and VEXT2, are communi-
cated to respective lines 258 and 260 within meter 5. How-
15 ever, these lines are not powered when the meter is on a base, but rather are powered independently and mutually exclusively of VLOGIC when the BAM contents are to be read independent of microcomputer 90. VEXT1 also communicates to lines 210 and 215 through respective diodes 262 and 263.
20 Similarly, VEXT2 communicates to lines 212 and 215 through respective diodes 265 and 266.

Since the voltage from regulator 242 exceeds the voltage of batteries 220 and 230, when VLOGIC is supplied to line 240, the regulated output of regulator 242 supplies
25 power on lines 210 and 212. This produces a level of +5 volt on lines 210 and 212, there being a voltage drop of approximately 0.7 volts across each of diodes 252 and 255. When there is no power from regulator 242, batteries 220 and 230 provide power to lines 210 and 212, respectively. Batteries
30 220 and 230 are preferably lithium primary cells having a shelf life typically in excess of ten years. Suitable batteries are manufactured by Mallory.

Two additional external power sources in the base are coupled to meter 5. A source of +34 volt, designated
35 VMOTOR, is coupled to a line 270 within meter 5 to supply power to stepping motors 45, 46, 47, and 48. A source of +5 volt, designated VDISP, is coupled to a line 275 within meter 5 to supply power for the digits of displays 15 and 17.



In addition to supplying power to meter 5, the base performs paper handling functions, and communicates signals to meter 5 in order to synchronize the activity within meter 5 with the paper handling that is ongoing. These signals are illustrated schematically in Fig. 6. A grounded connection in the base is communicated to a line 280 within the base to supply a signal designated MB, which is low whenever the meter is mounted to a base. A signal, designated TAPE is communicated to a line 282 within meter 5, a low level signifying to the meter that a postal tape is to be printed. A signal, designated ENVELOPE is communicated to a line 285 within meter 5, a low level signifying that an envelope is to be printed. A signal, designated REPEAT, is communicated to a line 287 within meter 5, a low level signifying that multiple tapes are to be printed with the same value. A signal, designated CLUTCH, is communicated to a line 290 within meter 5, a low level signifying the fact that a clutch within the motor in the base that provides mechanical power to the print head within the meter has been pulled. An external interrupt signal, designated EXTINT, is communicated to a line 292 in meter 5 indicating the need to service an I/O request.

From a functional point of view, the signals on lines 280, 282, 285, 287, 290, and 292 are treated like switches, the status of which is periodically determined by microcomputer 90 during execution of the foreground routine in order to properly allocate task flow.


Fig. 7 illustrates the generation of certain signals within meter 5. A signal, designated POMODE, is produced on a line 295 to reflect the status of mode changing switch 30. Mode changing switch 30 is magnetically coupled to a Hall Effect sensor 297 within meter 5. The normal state communicates a high level to line 297, indicating that the meter is in the "mail room mode". When switch 30 is placed in the "post office mode" position, sensor 297 communicates a low level to line 295.

The position of the print head is detected by a Hall Effect sensor 300 which cooperates with an iron element

301 on the printhead shaft. A signal, designated LEFT HOME, is produced on a line 302, the level on line 302 being high when the print head is in its home position and low when the print head is away from its home position.

5 Fig. 8 includes a circuit schematic of power surveillance circuitry 105. Power surveillance circuitry 105 has the function of producing a signal, designated PWRLSS, on a line 330, whose level is high whenever VLOGIC is below about 9.5 volts. Line 330 is coupled to an output 335 of a
10 voltage comparator 340. Comparator 340 is supplied at a first input 341 with VCC, and at a second input 342 with a precisely maintained fraction of VLOGIC. The fraction is defined by a voltage divider including a 5.9K, 1% resistor 345 and a 10K, 1% resistor 347. So long as VLOGIC remains
15 above approximately 9.5 volts, the voltage at input 342 remains above 5 volts. VCC, being a regulated output voltage based on VLOGIC, remains at 5 volts so long as VLOGIC remains above approximately 7 volts. When VLOGIC falls below 9.5 volts, the voltage at input 342 falls below the voltage at
20 input 341. Comparator 340 then causes a high level to appear at output 335 which is coupled to line 330. This PWRLSS signal is communicated to microcomputer.90 on line 330 to initiate a power loss interrupt.

 The relative sequence of these events is best
25 understood with reference to Fig. 10 which is a timing diagram illustrating a complete power up and power down cycle. As VLOGIC increases from 0 to 7 volts, VCC increases correspondingly from 0 to 5 volts. As VLOGIC increases from 7 volts to about 9.5 volts or above, VCC maintains its regulated 5 volt output. During the increase of VLOGIC from 0 to
30 7 volts, PWRLSS, which depends on VLOGIC which powers comparator 340, rises to assume a high level. Once VLOGIC has increased above 9.5 volts, PWRLSS assumes a low level, the transition being designated 355. Similarly, as VLOGIC falls
35 below 9.5 volts, PWRLSS assumes a high level, the transition being designated 358. PWRLSS remains high as VLOGIC decreases from 9.5 to 7 volts, and thereafter generally follows VLOGIC as it decreases towards 0. VCC remains at 5 volts so long as VLOGIC is above 7 volts.



Transitions 355 and 358 in the PWRLSS signal are significant events that initiate a particular sequential generation of MPCLR and SYSCLR signals. This insures that operation of the electronic control system, including micro-
5 computer 90, only occurs during periods when the available power is sufficient to ensure reliable operation of the components. Moreover, the sequence is designed to provide sufficient time to complete any updating of BAM registers before inhibiting microcomputer operations in order to main-
10 tain data integrity.

Broadly, SYSCLR, when high, inhibits operation of the print mechanism and writing data to BAMs 100 and 102. MPCLR, when high, inhibits operation of microcomputer 90. Accordingly, MPCLR must go low before SYSCLR goes low and
15 remain low until after SYSCLR goes high in order to ensure that microcomputer 90 maintains control at all times that printing could occur or data could be written to the BAMs.

Referring to Fig. 10, the requirements on the timing of MPCLR and SYSCLR can be seen. Once PWRLSS has gone
20 low, MPCLR must remain high for a duration 360 to ensure that microcomputer 90 has had power for a sufficient time to operate reliably. MPCLR then goes low, indicated schematically as transition 362. Microcomputer 90 then commences operation and SYSCLR, which has been high, then goes low, to
25 allow normal meter functioning. This transition, indicated 365, must occur later than transition 362 by an interval 370. Similarly, when PWRLSS goes high, SYSCLR must remain low for a sufficiently long time to complete any writing to the BAMs that is in progress. A duration 372 must elapse before
30 SYSCLR goes high, the transition being designated 375. After SYSCLR has gone high, MPCLR goes high, the transition being designated 377 occurring after transition 375 by an interval 380. The length of intervals 370 and 380 is of no particular significance, so long as the particular ordering is main-
35 tained. Intervals 360 and 372 are functions of the particular hardware configuration and components used. For the embodiment described herein, interval 360 is preferably greater than 50 milliseconds while interval 372 is greater

than 20 milliseconds. During interval 372 it is necessary that VCC remain at its 5 volt level to insure that the electronic components function reliably and that an enable on the BAMS (to be discussed below) remain above 2.2 volts. This means that VLOGIC cannot be allowed to fall below 8 volts in less than 20 milliseconds. This reserve capacity is provided by capacitor 201 (Fig. 5) which insures that enough power will be available to maintain reliable operation during interval 372.

Fig. 8 also includes a circuit schematic of system reset circuitry 110, which has the function of generating MPCLR and SYSCLR signals according to the above sequence in response to transitions 355 and 358 of the PWRLSS signal. System reset circuitry 110 actually provides complementary SYSCLR and MPCLR signals, designated $\overline{\text{SYSCLR}}$ and $\overline{\text{MPCLR}}$, on respective lines 390 and 392.

Circuitry 110 includes a comparator 400 having a first (positive) input 401 maintained at a fixed fraction of VCC by a voltage divider chain comprising 5.9K resistor 402 and a 10K resistor 404. A second (negative) input 406 is coupled to a capacitor 407 to ground and to a resistor 408 to VCC. The PWRLSS signal on line 330 communicates to comparator input 406 through a diode 409. An output 410 of comparator 400 communicates a circuit point 415.

Circuit point 415 is coupled through a resistor 417 to a circuit point 422 which communicates to different portions of the circuitry. First, circuit point 422 is coupled through a diode 425 to the base of a Darlington transistor 428, the emitter of which drives line 390 ($\overline{\text{SYSCLR}}$). The base of transistor 428 is coupled to a capacitor 430 to ground, and to VCC through a resistor 432, and further to the collector of a transistor 435. Second, circuit point 422 is coupled through a diode 440 to an intermediate circuit point 442 of a chain defined by a 75K resistor 445 to VCC, and a 4.7 microfarad capacitor 450 to ground. Capacitor 450 is coupled to the base of a Darlington transistor 455. The collector of transistor 455 is coupled through a diode 457 to a circuit point 458. Circuit point 458 is coupled through a

resistor 460 to the base of transistor 435, and through a resistor 465 to the base of a transistor 470, the collector of which is coupled to line 392 (\overline{MPCLR}).

Consider first a condition wherein a high level has persisted for a substantial length of time on line 330. Capacitor 407 is fully charged to VCC, and output 410 of comparator 400 is low. Accordingly, circuit point 415 is low, so that circuit point 422 is low. This causes the base of transistor 428 to be low, thereby causing transistor 428 to be in a nonconducting state so that the level on line 390 is low. That is, SYSCLR is high. The low level on emitter 422 maintains circuit point 442 at a low level, thereby keeping transistor 455 in a nonconducting state. Accordingly, circuit point 458 is high so that transistor 435 assumes a conducting state. The collector of transistor 435 is thus at a low level, thereby reinforcing the low level imposed through diode 425 across capacitor 430. Also, the high level at circuit point 458 turns transistor 470 on, and causes a low level to appear on line 392 (\overline{MPCLR} is true).

When PWRLSS goes low (transition 355), the following events occur in order to cause \overline{MPCLR} and SYSCLR to go low in the required sequence producing intervals 360 and 370. When the level on line 330 goes low, diode 409 conducts, thereby discharging capacitor 407 and causing the input 406 of comparator 400 to go low. Output 410 goes high, causing circuit point 415 to go high. A high level thus appears at circuit point 422. This blocks diode 440, thereby causing capacitor 450 to become charged through resistor 445. Resistor 445 is relatively large (75K) thereby resulting in a substantial delay before the voltage across capacitor 450 is enough to switch transistor 455 to its conducting state. This delay substantially defines interval 360, since once transistor 455 is conducting, circuit point 458 goes low, thereby turning off transistor 470 and causing a high level to appear on line 392. That is, \overline{MPCLR} goes low. Once the delayed low level has appeared at circuit point 458, transistor 435 is turned off, thereby allowing capacitor 430 to become charged through resistor 432. This causes transistor

428 to become conducting after a time interval defined by capacitor 430 and resistor 432, at which point, the level on line 390 goes high. That is, SYSCLR goes low.

When PWRLSS has been low for some time so that the various components of power surveillance circuitry 110 are in the states described above, and then PWRLSS goes high (transition 358), the following sequence of events occurs. The high level on line 330 blocks diode 407, causing capacitor 408 to become charged through resistor 409. After a delay in excess of 20 milliseconds, the voltage on capacitor 408 exceeds the fractional level defined by resistors 402 and 404, and output 410 of comparator 400 goes low. This delay establishes interval 372. A low at output 410 causes a low at circuit points 415 and 422. This causes the base of transistor 428 to go low, thereby turning off transistor 428 causing a low to appear on line 390. That is, SYSCLR goes high.

When circuit point 422 goes low, diode 440 which was formerly blocked becomes conductive, causing capacitor 450 to discharge through resistor 447. After a delay determined by the time constant of capacitor 450 and resistor 417, transistor 455 is turned off thereby causing its collector to go high and conduction to occur through diode 457. This causes transistor 435 to conduct, thereby reinforcing the low at the base of transistor 428. In addition, the high level at circuit point 458 causes transistor 470 to conduct, thereby causing a low level to appear on line 392. That is, MPCLR goes high.

Fig. 9 is a circuit schematic of circuitry for generating signals to inhibit meter functioning upon detection of a fault condition. Two redundant circuits are employed, and with an exception to be discussed below, these two circuits are equivalent. Thus only one will be discussed herein.

One-shot 113 has an input 500 which is responsive to a falling edge. A resistor 506 and a capacitor 504 are coupled to one-shot 113 to establish basic timing. One-shot 113 has a reset input 507 to which the $\overline{\text{MPCLR}}$ signal on line

392 is coupled by a line 508. One-shot 113 has a complementary output 509 which is coupled to a first input of a NAND gate 510. NAND gate 510 has a second input to which is coupled the SYSCLR signal on line 390 by a line 511. NAND
5 gate 510 has an output 512 which is coupled to flip-flop 111. Flip-flop 111 has an output 515 which is coupled through a diode 517 to the base of a Darlington transistor 520, the collector of which is coupled to line 390 of system reset circuitry 110 by line 511. Under normal conditions, micro-
10 computer 90 periodically causes the generation of a signal, designated MPTS1, to fire one-shots 113 and 114. As will be discussed below, the signal from microcomputer 90 causes a low level to appear at input 500. A low level is generated on complementary output 509 in response to a negative going
15 level at input 500, and remains low for a duration defined by capacitor 504 and resistor 506. In the preferred embodiment this duration is 20 milliseconds. Since one-shot 113 is a retriggerable one-shot, a subsequent signal at input 500 extends the period that output 509 remains low. Thus,
20 repeated signals at less than 20 millisecond intervals maintain output 509 low. So long as the level on output line 509 is low, the level on line 512 is high. Fault flip-flop 111 maintains a low level on output line 515 so long as the level on line 512 is high. If, for some reason, the level on line
25 512 goes low, a high level appears on flip-flop output 515, and remains, even if the level on line 512 goes high again. A high on line 515 blocks diode 517 and causes a high level to appear at the base of transistor 520. This causes a low level to appear on line 511, which low level causes the level
30 on line 390 (SYSCLR) to remain low regardless of the level of PWRLSS. Accordingly, once fault flip-flop 111 has been set to have a high level at its output 515, a low level is maintained on line 390. That is, once fault flip-flop 111 is set, SYSCLR remains high regardless of power up and power
35 down cycling. By coupling MPCLR to one-shot reset input 507, the level on one-shot output 509 is caused to remain high after MPCLR is false until signals from microcomputer 90 appear at input 500. Also, by coupling SYSCLR to NAND

gate 510, a high level is maintained on line 512 when SYSCLR is true. Thus, fault flip-flop 111 is prevented from being set to its "faulted state" during the times that SYSCLR is true during normal power cycling.

5 Fault flip-flop 112 and retriggerable one-shot 114 are coupled in a similar way with the exception that one-shot 114 uses the non-complementary output on a line 540. The output on line 540 is coupled to both inputs of a NAND gate 542 which is physically removed from the integrated circuit
10 chip including one-shots 113 and 114. Accordingly, if the chip on which one-shots 113 and 114 are located failed in a way that would cause all outputs to assume a high or low level, one of the fault flip-flops would be set. Only a dual failure which would cause the output on line 509 to remain
15 low and the output on line 540 to remain high would go undetected. This is in keeping with the general meter design which can tolerate one failure and maintain integrity, but cannot accommodate certain combinations of compensating failures.

20 The level on line 390 is communicated to a circuit point 549 through cascaded Darlington buffers 545 and 547. Circuit point 549 is coupled to the cathode of fault indicator light 20, preferably a light emitting diode, the anode of which is coupled to VCC. Thus when circuit point 549 goes
25 low light 20 is illuminated. Thus, light 20 is briefly illuminated during power up and power down cycling for those periods that circuit point 549 is low, providing the operator with an indication that the light works. Once SYSCLR has gone high, light 20 is illuminated. This occurs when a fault
30 condition has occurred, regardless of power cycling.

Figs. 11a, 11b, and 11c, taken together, are a circuit schematic of microcomputer 90, BAMS 100 and 102, and circuitry for generating and responding to control signals related to accessing the BAMS.

35 Of the various control signals discussed above, microcomputer 90 reacts directly to PWRLSS and $\overline{\text{MPCLR}}$. A high level on line 330, signifying a low power condition is inverted by a NOR gate 560 and communicated to an interrupt



input 562 of microcomputer 90. A low level appearing at interrupt input 560 causes a power loss interrupt as discussed in connection with Fig. 8. The signal \overline{MPCLR} on line 392 is communicated to a reset input on microcomputer 90. A
5 low level on line 392 ($MPCLR$ true) causes microcomputer 90 to be disabled and to cease execution.

Microcomputer 90 also generates control signals for selecting which of BAMs 100 and 102 is to be accessed, and which of I/O expanders 92 and 93 is to be accessed. These
10 two selection procedures make use of the same data lines 132 and 133 (lines P24 and P25 of bus 130). The significance of the signals on lines 132 and 133 depends on whether microcomputer 90 is executing a background routine (BAM select) or the foreground routine (I/O expander select).

15 Microcomputer 90 selects one of I/O expanders 92 and 93 by causing the generation of signals designated PROG1 and PROG2 on respective lines 565 and 567. The level on output lines 132 and 133 is communicated to respective OR gates 570 and 571, the other input of each of which is connected to an output line 572 of microcomputer 90. The level
20 on line 572 is normally high, but goes low at the same time that a control code for an I/O expander appears on line 134. The level on line 572 then goes high and at the same time data to be transferred appears on line 134. Thus the signal
25 on line 572 appears on line 565 or 567, depending on whether line 132 or 133 is low.

Reading and writing to BAMs 100 and 102 are controlled by providing appropriate levels at four control inputs of each BAM. Each BAM includes two chip enable inputs
30 573 and 574, designated $\overline{CE1}$ and $CE2$, respectively. The level at chip enable input 573 must be low to enable access to the BAM contents, while the level at chip enable input 574 must be high. A low level at chip enable input 574 causes the BAM to assume a so called "low power mode" in which it
35 draws only the small amount of power required to maintain its contents. In the low power mode, access to the BAMs is inhibited. Each BAM includes an output disable input 575, designated OD, which must be at a low level in order to read

the BAM. Each BAM includes a write enable input 576, designated R/\bar{W} , the level at which must be low for writing to occur, and high to disable writing during a read operation.

Microcomputer 90 selects which of BAMs 100 and 102 is to be read by causing a low level to appear on either of lines 132 and 133, which low level is communicated to respective chip enable inputs 573. Microcomputer 90 controls reading and writing to the BAMs according to the output level on a line 579 (line P27 of data bus 130). Line 579 communicates to a first input of a NAND gate 580, and the signal \overline{SYSCLR} on the line 390 is communicated to a second input of NAND gate 580, so that the output of NAND gate 580 is low only when \overline{SYSCLR} is high and the output on line 579 is high. The signal at the output of NAND gate 580 is communicated on a control line 585 to control inputs 576 on BAMs 100 and 102 to control writing to both BAMs.

The levels VLOGIC, VEXT1, and VEXT2 control reading and writing to the BAMs independent to microcomputer 90. VLOGIC communicates to respective chip enable inputs 574 on BAMs 100 and 102 through a common back biased 5.1 volt zener diode 590 and respective diodes 592 and 595. VEXT1 and VEXT2 also communicate to the respective inputs 574 through respective diodes 597 and 600. Thus when VLOGIC rises above about 5.7 volts, zener diode 590 breaks down and diodes 592 and 595 conduct. Thus the level at inputs 574 begins to rise and when it reaches about 1.5 volts, puts BAMs 100 and 102 into their so-called "high power mode" in which reading and writing may occur so long as other control signals are present. In the absence of VLOGIC, powering VEXT1 or VEXT2 allows the BAMs to be accessed. In the absence of sufficiently high levels of VLOGIC, VEXT1, and VEXT2, chip enable inputs are held low by pull-down resistors to ground.

Writing to the BAMs can occur independent of microcomputer control 90 for the purpose of allowing a postal service employee to zero certain locations when the meter has faulted. To permit this to occur, the signal \overline{POMODE} on line 295 and a signal \overline{TWRITE} on a line 609 are communicated to first and second inputs of a NOR gate 610. The

output of NOR gate 610 is communicated to the base of a transistor 615, the collector of which is coupled to BAM control line 585. Therefore, a low level on lines 295 and 609 causes a high level at the output of NOR gate 610, there-
5 by turning transistor 615 on and causing a low level to appear on line 585, thereby allowing access to the BAMs.

In addition to controlling reading and writing to the BAMs, the SYSCLR signal on line 390 cooperates with signals from microcomputer 90 in order to generate a signal
10 designated SON on a line 620 for controlling print head solenoid 85. Signals on an output line 622 (line P26 of bus 130) are communicated to a first input of a NOR gate 625. The signal on line 390 (SYSCLR) is first inverted by a NOR
15 gate 628, to produce a SYSCLR signal and communicated to a second input of NOR gate 625, the output of which is coupled to line 620. Therefore, a high level on line 390 and a low level on output line 622 are required for SON to be high. Otherwise, SON is low. As will be discussed below, SON must be high to allow the print head to rotate. The SYSCLR signal
20 from NOR gate 628 is communicated to motor interface circuitry on a line 629.

Fig 12a is a circuit schematic of circuitry for interfacing microcomputer 90 to the switches, displays, BAMs, and motor position sensors. Data flow is established through
25 I/O expander 92. I/O expander 92 receives signals on four bit data line 134 (lines P20 -P23 of data bus 130), in order to determine which of four data ports 650, 652, 654, and 656 is to be accessed, whether a read or a write operation is to be carried out, and whether the data is to be logically
30 manipulated. Once this selection has occurred, data is communicated to or from microcomputer 90 on the same line 134. In the preferred embodiment, I/O expander 92 is used for generating address codes at its I/O ports in order to control demultiplexers 94 and 95 and multiplexers 96 and 97.
35 Data port 650 is coupled to an address input 660 of multiplexer 96. Multiplexer 96 has 16 data inputs, not all of which are used. The following signals, the generation and significance of which were discussed in connection with Figs.

6 and 7 are communicated to respective inputs of multiplexer 96: $\overline{\text{POMODE}}$ on line 295; $\overline{\text{LEFT HOME}}$ on line 302; $\overline{\text{MB}}$ on line 280; $\overline{\text{TAPE}}$ on line 282; $\overline{\text{ENVELOPE}}$ on line 285; $\overline{\text{REPEAT}}$ on line 287; $\overline{\text{CLUTCH}}$ on line 290, and $\overline{\text{EXTINT}}$

5 on line 292. Selector switch 27 is coupled to two inputs on multiplexer 96 in order to communicate a low level to that input corresponding to the position of switch 27.

In addition to the above signals, signals designated $\overline{\text{KIN}}$ and $\overline{\text{RIN}}$ on respective lines 670 and 675 are
 10 communicated to respective data inputs of multiplexer 96. $\overline{\text{KIN}}$ and $\overline{\text{RIN}}$ are time multiplexed signals reflecting the state of selector switch 25 and keyboard 12. These will be discussed below.

Data port 652 is coupled via four-bit line 135 to
 15 an address input of multiplexer 97 and on a parallel line to BAMS 100 and 102. Data ports 654 and 656 communicate with demultiplexers 94 and 95 on respective data lines 680 and 685 for refreshing displays 15 and 17, and reading selector 25 and keyboard 12, as will be discussed presently. These
 20 operations cause the generation of $\overline{\text{KIN}}$ and $\overline{\text{RIN}}$ on lines 670 and 675.

Fig. 12a is a circuit schematic illustrating circuitry for periodically energizing the digits of displays 15 and 17, and reading keyboard switches 38, 40, and 42, and
 25 selector switch 25. Each digit of nine-digit display 17 comprises eight light emitting diode segments including seven bars for representing numeric symbols, and a decimal point. Display 17 includes a digit select input 695 for each digit (nine in all), and eight segment select inputs 698. Display
 30 17 is a common cathode device, which means that a given digit select input 695 is coupled to the cathodes of all eight segments for that digit, while a given segment select input is coupled to the anodes of a particular segment for all nine digits. Thus, a low level at a particular digit select input
 35 695 causes that digit to be illuminated according to which of segment select inputs 698 is at a high level.

Demultiplexer 94 includes ten data outputs 700, and a four-bit select input 702, such that a low level appears at



the output 700 corresponding to the binary code defined by the levels at select input 702. Nine of outputs 700 are connected in a one-to-one fashion to digit select inputs 695 on nine-digit display 17. Each of the ten demultiplexer
5 outputs 700 is coupled to one pole of a numeric keyboard switch 38, the other pole of which is coupled to line 675 \overline{RIN} . Buffered data on line 129 is communicated through current limiting resistors 705 to a corresponding eight-bit line 707. Each individual conductor is coupled to one of
10 segment select inputs 698 with one of the conductors, designated 708, coupled to the segment selector controlling the decimal points. Thus, according to the binary code on four bit line 680, one of multiplexer outputs 700 is set to a low level, which causes a low level to appear on line 675 if the
15 corresponding keyboard key 38 was closed. Additionally, in the case of those nine of outputs 700 which are coupled to display digit select inputs 695, a low level causes that display digit to be illuminated according to the levels on line 129. Thus, each time one of the digits is energized, a
20 corresponding signal indicates the status of a particular keyboard key. After the nine digits have been energized, the tenth demultiplexer output is selected, providing a level on line 695 that indicates the status of the tenth numeric keyboard key 38.

25 Four-digit display 15 comprises first and second two-digit displays 720 and 722, each digit of each display having eight light emitting diode segments including seven bars and a decimal point. Each digit includes a digit select input 730 and eight segment select inputs. For the particu-
30 lar display components in the preferred embodiment, each digit is independent, thereby requiring individual segment selection. Data on eight-bit line 707 is communicated to the segment selector inputs. Digit displays 720 and 722 are common anode devices, so that when the level at one of digit
35 selector inputs 730 is high, that particular digit is illuminated according to which of the individual lines of line 707 is at a low level. Conductor 708 of eight-bit line 707 is coupled to the decimal point segment selectors for display

720, but two-digit display 722, corresponding to the least significant digits, has no connection to line 708. Rather line 708 is coupled to the cathode of indicator light 22, preferably a light emitting diode, the anode of which is
5 coupled to that digit selector input 730 corresponding to the least significant digit of display 15.

Demultiplexer 95 has a four-bit select input 750, and ten data outputs, nine of which, designated 751 - 759, are connected as follows. Data outputs 751 and 752 are
10 coupled to respective first poles of keyboard switches 40, and 42, respective second poles of which are coupled to line 675. Output 758 is coupled to a line 765 to provide a signal, designated \overline{EIO} for controlling I/O peripheral devices 131. Output 759 is coupled to a line 767 to generate the
15 signal $\overline{MPTS1}$ which is communicated to input 500 of fault one-shots 113 and 114.

Each of data outputs 753 - 757 is coupled to a contact of five position display selector switch 25, the common wiper of which is coupled to line 670. Of the five
20 data outputs 753 - 757, four of them, 753 - 756, are coupled in a one-to-one fashion to digit select inputs 730. However, the coupling is not direct as in the case of the circuitry for energizing display 17, since display 15 is a common anode device. Each digit selector input 730 is coupled to the
25 collector of a PNP transistor 765, the emitter of which is held at VDISP. Each of data outputs 753 - 756 is resistively coupled to the base of a corresponding one of transistors 765.

Accordingly, depending on the binary code defined
30 by the levels on four bit line 685, a particular one of data outputs 751-759 is provided with a low level. When output 751 or output 752 goes low, a low level is applied to line 675 if special keyboard key 40 or 42, respectively, is depressed. If either of outputs 758 or 759 is selected, a
35 signal having a falling edge followed by a low level is generated on line 765 or 767, respectively, communicating to other portions of the circuitry as described above. A low level on any of outputs 753 - 756 causes the corresponding



transistor 765 to become conducting, thereby providing a high level at the particular digit select input 730. Depending on which of the individual lines of line 707 is at a low level, the particular segments of the selected display digit will
5 light. Noting that neither of the least significant digits ever requires a decimal point following, indicator light 22 is connected in a manner that substitutes for the decimal point of the least significant digit whose decimal point cathode is not connected to anything. Thus, when the least
10 significant digit is selected and the level on line 708 is low, "Add \$" indicator light 22 is energized. Also, data outputs 753 - 756, in cooperation with data output 757, provide the signal \overline{KIN} for communicating the status of switch 25 to microcomputer 90.

15 Fig. 13 is a circuit schematic illustrating the circuitry for interfacing microcomputer 90 to stepping motors 45 - 48 and position sensors 75 - 78. While Fig. 13 illustrates the complete interface circuitry, only one stepping motor 47 and corresponding position sensor 77 is shown for
20 clarity. Broadly, the interface circuitry includes I/O expander 93, multiplexer 97, and a plurality of Darlington buffers 780 coupled to VMOTOR to provide buffered outputs that are either low or at a level VMOTOR. I/O expander 93 includes four data ports 785, 786, 787, and 788, signals at
25 which are communicated through Darlington buffers 780 to respective motors 45, 46, 47 and 48, with port 787 communicating with motor 47. I/O expander 93 has an enable input 790 to which SYSCLR is communicated via line 629, and a program input 792 to which PROG1 is communicated on line 133.
30 Control signals and data are communicated in a time multiplexed fashion on four-bit data line 134.

Motor 47 may assume one of ten positions over its angular range that correspond to proper print wheel settings, and eleven positions halfway between. Thus the possible
35 positions can be broken into four groups including a group of so-called "odd" positions, a group of so-called "even" positions, and two groups of so-called "half" positions, each position of which is halfway between an even and an odd position.

Motor 47 includes four windings, corresponding to the four groups of positions. In general, the motor can only be stepped to an adjacent position by energizing the winding corresponding to the group of which the adjacent position is a member. One side of each winding is coupled to VMOTOR in the meter base through a current limiting resistor 789 located in the base. The other side of each winding is coupled to one of the buffered outputs corresponding to the data lines from port 787. A low level on a line at data port 787 turns off the buffered output, thereby resulting in no voltage across the particular winding. Only when a high level appears on an output line at port 787 is a low level communicated to the motor winding, thereby energizing it.

Since energization of a motor winding generally only causes the motor to step to an adjacent position, and since only four types of positions are recognized, absolute position determination may only be established by first driving motor 47 to one of its end points, and thereafter keeping track of all step commands. This is done during execution of the initialization routine.

Position sensor 77 comprises four sets of contacts 791, 792, 793, and 794, each set including ten sequentially spaced contacts, and four grounded wipers 801, 802, 803, and 804, each associated with a respective contact set. Wipers 801 - 804 are mechanically coupled to the mechanical output of motor 47, so that at a given integral position of motor 47, each of wipers 801 - 804 makes electrical connection with one of the contacts in its respective contact set 791 - 794. A binary code representative of the position of wipers 801 - 804 is generated on four corresponding output lines 811, 812, 813 and 814, one end of each output line being coupled only to particular individual contacts within its respective contact set. For example, the "1" bit is generated on line 811 by coupling line 811 to every other contact in set 791; the "2" bit by coupling line 812 to every other group of two contacts in contact set 792. The other end of each contact line is coupled to a data input of multiplexer 97. Four-bit data line 135 from port 652 of I/O expander 92 is coupled to

an address input 820 of multiplexer 97. According to the binary code represented by the levels on line 135, one of the four bits of information for one of the four position sensors appears at the output of multiplexer 97, and is communicated
5 on line 99 to microcomputer 90.

Print head solenoid 85 prevents movement of the print head away from its home position unless the solenoid is energized. Control is established by the signal SON on line 620. Solenoid 85 includes input terminals 824 and 825.
10 Input terminal 824 is coupled to VMOTOR in the base through current limiting resistors 827 and 828 located in the base. A capacitor 829 is connected across resistor 828 to provide greater initial current. The signal SON is communicated to the base of a Darlington transistor 832, the emitter of which
15 is grounded and the collector of which is coupled to solenoid input terminal 825. Thus, a high level on line 620 causes a low level to appear at input terminal 825 of solenoid 85, thereby energizing the solenoid to permit movement of the print head. A diode 835 is connected across solenoid 85 to
20 protect Darlington transistor 832 from overvoltage when SON goes low.

In view of the above description covering the general operation and detailed construction of meter 5, a description of the detailed operation of microcomputer 90 may
25 be understood. The following discussion and the accompanying figures provide additional detail with respect to the discussion with reference to Fig. 4 outlining the general operation of microcomputer 90.

The operating program for microcomputer 90 is
30 stored in 2,048 bytes of ROM while working memory is provided by 64 bytes of random access memory (hereinafter RAM). Fig. 14 illustrates in tabular form a suitable RAM allocation for storing information required by the operating program of microcomputer 90. For ease and consistency of nomenclature,
35 the individual memory locations will be referenced in first instance with respect to their hexadecimal address within the 64 bytes of RAM, and subsequently by reference to appropriate mnemonics.

Location 0-7 Hex, designated R0-R7 and locations 18-1F Hex, designated R0'-R7' are two sets of directly addressable working registers.

Registers R0-R7 are allocated to the background tasks, to be described more fully below. Register R2 is dedicated to the background status and is hereinafter designated BAKSTA. The individual bits are shown schematically in Fig. 16, and include the power loss bit and the various WAKEUP bits set by the foreground to signify to the background dispatcher loop that a particular task is to be carried out. The particular significance of the various WAKEUP bits will be discussed with reference to the foreground routine and the appropriate background tasks described below. The individual bits in BAKSTA are sequentially checked by the background dispatcher loop as described above with reference to Fig. 4.

Locations 34 Hex and 35 Hex, designated KEYSTA and PRSTA respectively, are used to designate status for the keyboard and printer tasks. Locations 38-3B Hex, designated STEPIN(0)-STEPIN(3) and locations 3C-3F Hex, designated STEPTK(0)-STEPTK(3) provide an additional data base for the print routine. Location 36 Hex, designated PRCTR, is used to store a counter which, when set to a non-zero value, is decremented at each entry into the foreground routine in order to keep track of the time that has elapsed since various print subtasks were initiated. Locations 8-13 Hex provide 12 bytes for a six level stack to allow for six levels of sub-routine calls.

As discussed above, the foreground routine is executed in response to periodic timer interrupts, that occur at 400 microsecond intervals. The main functions of the foreground routine are energizing the digits of four-digit display 15 and nine-digit display 17, ascertaining the status of various switches and signals (collectively referred to as switches), debouncing portions of the switch information, and triggering one-shots 113 and 114. These functions are carried out by selecting I/O expander 92 and setting up appropriate binary codes on four-bit line 134 to communicate with

the appropriate switch or display digit via demultiplexer 94, demultiplexer 95, or multiplexer 96.

Registers R0'-R7' are the working registers used by the foreground that enable the foreground routine to carry out its functions according to the desired sequence. For example, four-digit display 15 is sequentially energized four times for every time that nine-digit display 17 is energized in order to provide adequate digit intensity. Register R3' is a counter for keeping track of this sequencing. Register R2' is a pointer that indicates which display digit is currently being done, while R1' is a basic pointer.

Locations 20-28 Hex and 30-33 Hex are used to store the display segment codes for the current values of the digits to be displayed on nine-digit display 15 and four-digit display 17, respectively. The foreground routine sequentially places these segment codes on bus 125 (P1) as the digits are sequentially selected through demultiplexers 94 and 95.

Switch information is stored in four switch registers. Location 1F Hex, designated R7FORG, is dedicated to certain of the base synchronization signals and other signals received through multiplexer 96, the status of which is desired without debouncing. Location 2A Hex, designated REG07, location 2C Hex, designated REG89, and location 2E Hex, designated REGSW, reflect the current debounced state of all the switches that can be set by the operator. The respective bit allocations for switch registers R7FORG, REG07, REG89, and REGSW are shown in Fig. 15. Locations 2B Hex, 2D Hex, and 2F Hex are so-called LAST TIME CHANGE registers corresponding to switch registers REG07, REG89, and REGSW, respectively and carry information indicating which switches corresponding to the bits of the corresponding switch register had changed status on the previous reading by the foreground.

As discussed above, switches are debounced in order to eliminate the effects of switch noise, and the status of a switch to be debounced is considered to have changed only if such change has persisted for two successive readings.

Debouncing of the switches is done in two steps. Preliminary debouncing is done on a switch-by-switch basis as each switch is read while final debouncing occurs when all the switches in a given switch register have been read and preliminarily
5 debounced. As discussed above, nine of the ten keyboard switches 38 and four of the five positions of register display selector switch 25 are each read at the same time that a particular display digit is energized. Thus, each one of these switches is read on a separate pass through the foreground routine in conjunction with the energization of a
10 particular display digit. Other switches are read serially in groups during particular single passes through the foreground routine.

Preliminary debouncing occurs as each switch is
15 read. During preliminary debouncing, the switch register for the group of switches being debounced is copied to working register R4', and the corresponding LAST TIME CHANGE register is copied to working register R5'. R0' points to the current one of the switch registers that is being debounced. Preliminary
20 debouncing on the group of eight switches in the particular register stored in R4' occurs over a number of foreground entries. As each switch is read on line 98 from multiplexer 96, the switch state is compared with the corresponding bit in the debounced switch register currently
25 copied into register R4'. A corresponding bit is set in register R6' if the switch state read represents a change from the debounced value in register R4'. When all eight switches in the register being debounced have undergone this preliminary debouncing, R6' contains a record of those
30 switches whose state at the most recent reading has changed with respect to the debounced values. This is stored as the LAST TIME CHANGE register for the next pass of the foreground. Final debouncing occurs by comparing registers R5' and R6'. If, for a particular switch, a change has
35 persisted for two successive readings (corresponding bits in registers R5' and R6' being set) the corresponding bit in the debounced switch register is set to reflect the newly debounced state.

Fig. 17 is a flow chart of the foreground routine illustrating logical branches for executing a preferred sequence for servicing displays 15 and 17, ascertaining the status of the various switches, and performing debouncing.

5 Upon the occurrence of a timer interrupt, microcomputer 90 enters the foreground routine and performs entry tasks generally designated as block 850. These include saving the background accumulator, restarting the 400 microsecond timer, and checking the value of PRCTR (the printer counter). If
10 PRCTR is non-zero, PRCTR is decremented by 1 and the TIMER WAKEUP bit is set if PRCTR has reached zero. The state of bus 130 is saved in register R1', and lines 132 and 133 are set up to select I/O expander 92 for the purpose of reading switches.

15 The routine then checks whether the foreground is enabled (logical branch 852). If the foreground is disabled, the routine triggers the one-shots by setting the appropriate binary code on demultiplexer 95 in order to select output line 767 (block 855). The routine then restores the back-
20 ground bus 130 and accumulator and returns (block 857).

If the foreground is enabled, the particular display digit is energized according to a preferred sequence. According to the preferred sequence, the digits of nine-digit display 17 are sequentially energized on successive entries
25 to the foreground routine, and then the digits of four-digit display 15 are sequentially energized. The energization of display 15 occurs four times for every time display 17 is energized. In order to maintain this sequence, the foreground routine maintains bookkeeping information in fore-
30 ground working registers R0'-R3'. This bookkeeping information includes pointers for addressing the particular switch register being debounced, and counters for determining which digit in a given display is to be energized and which of the four passes through the four digits display is being carried
35 out.

If at logical branch 852 the foreground is found to be enabled, the routine checks which display is being serviced (logical branch 860). If the nine-digit display is

currently being serviced, the routine checks if either the first or ninth keyboard key is to be read (branch 862). If so, the routine sets up registers R4' and R5' with REG07 or REG89, respectively, and the associated LAST TIME CHANGE register (block 865). Selecting one of the first nine keyboard keys ("0" through "8") to be read occurs in connection with the energization of one of the nine digits of display 17 (block 867). Reading of the tenth keyboard key ("9") is done independently of a digit energization.

5 During the first eight passes, the numeric keys corresponding to "0" through "7" are being sequentially read, and working register R6' accumulates information relating to changes in these switches for REG07. On the eighth pass (corresponding to keyboard key "7"), final debouncing of

15 REG07 occurs (block 872). A change in REG07 is flagged by setting bit 7 of register R7FORG. The foreground routine then prepares for debouncing switch register REG89 on subsequent foreground passes (block 873), and returns to the background (block 857).


20 If at logical branch 870 it is ascertained that the eighth keyboard key ("7") is not to be read, a separate test is made to determine whether the tenth key ("9") is to be read (logical branch 875). If the tenth key is not being selected, as would be the case most of the time, i.e. either

25 during preliminary debouncing of REG07 or of the ninth key for REG89, the pointer in working register R2' is incremented (block 877), and microcomputer 90 returns to the background.

 If at logical branch 875 it is determined that the tenth key is to be read, that key is read, and the additional

30 switches for REG89 are sequentially read on the same foreground pass and preliminarily debounced (block 880). REG89 is then finally debounced (block 882). The foreground routine then sets bit 7 of REG89 if the meter is not a fractional cents meter or if switch 30 is set to the "post

35 office" mode position or if switch 27 is in the .00 position, in order to provide information for the keyboard task (block 885). If a change has occurred to REG89 or REG07 (as remembered by R7FORG(7)), the KEYBOARD WAKEUP is set (block 887).



The foreground routine then reads those signals that are to be stored in register R7FORG (block 890). If EXTINT is set, the I/O WAKEUP is set (block 892). In the event that the newly read LEFT HOME or CLUTCH signal is different from the
5 corresponding formerly read signal, the HOME/CLUTCH WAKEUP is set (block 895). The pointers are then set so that the next entry into the foreground routine will cause servicing of four-digit display 15 (block 897).

When it is ascertained at logical branch 860 that
10 four-digit display 15 is to be energized, a test is made as to whether the first position of display selector switch 25 is to be read (logical branch 900). If so, the switch registers are set up to debounce register REGSW (block 865). Selecting one of the first four of the five positions of
15 switch 25 to be read occurs in connection with the energization of one of the four digits of display 15 (block 902). Reading the fifth switch position is not accompanied by selection of any of the four digits on the display. If the fifth position is not being read, the routine branches at
20 logical branch 905 to block 877 and then returns to the background. On the fifth position, the routine tests whether the four-digit display has been energized four times (logical branch 907). If not, register R2' is reset so that the next pass will read the first position (i.e. select first digit)
25 (block 908). The routine then branches to test whether the foreground is enabled (branch 852). The foreground will be found to be enabled (having previously been found so on the same foreground pass), so the first position is read on this pass as the routine branches as described above. If it is
30 ascertained that the display has been energized the required four times, the fifth position is preliminarily debounced and the remaining switches whose bits make up REGSW are read and preliminarily debounced (block 910), and register REGSW is finally debounced (block 912). If REGSW has changed, the
35 TAPE/ENVELOPE WAKEUP is set if the change is to either of the bits corresponding to the TAPE or ENVELOPE signals (block 915). The REGISTER DISPLAY WAKEUP is set if the change is to one of the register select switches (block 920). After

performing bookkeeping to prepare for servicing nine-digit display 17 on the next foreground entry (block 922), the routine then fires the one-shots (block 855) and returns to the background.

5 Fig. 18 is a memory map showing a preferred organization of BAMs 100 and 102, hereinafter sometimes referred to as BAM1 and BAM2, respectively. BAM1 and BAM2 are organized wherein a given hexadecimal address refers to a given hexadecimal digit (4 bits, hereinafter referred to as a nibble).

10 The key feature in a postage meter, electronic or mechanical, is the maintenance of accounting information to insure that all postage printed is paid for. As discussed above, microcomputer 90 maintains a descending register indicating the amount of postage still remaining (generally designated DR), an ascending register indicating the amount of postage printed (generally designated AR), and a control total which should remain constant between additions of postage at the post office (generally designated TOTAL). In keeping with the fault tolerance aspects of the present invention, these registers, redundant among themselves, are stored in redundant BAMs 100 and 102, the dual copies being designated DR1, AR1, and TOTAL1 for BAM1 and DR2, AR2, and TOTAL2 for BAM2, respectively.

25 Additional accounting information for use by the user is maintained in dual redundant user resettable registers designated BATCH COUNT and BATCH TOTAL. Each time postage is printed, these registers are respectively incremented by 1 and by the amount of postage printed. Resetting of these registers is effected by pressing "clear batch" key 42 simultaneously with "clear keyboard" key 40. A register designated COUNT is incremented by 1 on each print cycle to provide maintenance information for repair personnel at the factory by indicating the total number of print cycles the meter has undergone. The COUNT register is not accessible to the user.

35 DR1, AR1, and TOTAL1 are stored at nibbles 00-09 Hex, 10-19 Hex, and 20-29 Hex, respectively in BAM1, with DR2, AR2, and TOTAL2 being stored in corresponding locations

in BAM2. BATCH TOTAL 1 & 2, BATCH COUNT 1 & 2, and COUNT 1 & 2 are stored at nibbles 30-39 Hex, 40-49 Hex, and 50-59 Hex, respectively in BAM1 and BAM2. Register display selector switch 25 allows the user to display on nine-digit display 17 one of the registers DR, AR, TOTAL, BATCH TOTAL, and BATCH COUNT. A keyboard register, designated KB, is stored at nibbles 60-69 Hex.

Temporary copies of DR1, AR1, designated TDR1, and TAR1, respectively are stored in nibbles 70-79 Hex and 80-89 Hex, of BAM1. Corresponding copies designated TDR2 and TAR2 are stored in corresponding locations in BAM2. However, as will be described below, only TDR1 and TAR1 are updated during keyboard entry. Nibbles 90-99 Hex in each of the BAMS are used as a temporary register, designated TEMP1 and TEMP2.

Each of the registers described above contains ten nibbles, nine nibbles of which contain the numerical value in BCD representation. The tenth nibble of each register is called the dirty register nibble, bit 3 (most significant bit) of which is set to "1" before a register to register move is carried out into that register. The dirty register nibble is then zeroed at the end of the move.

All the registers except COUNT contain the information to three decimal places. Thus incrementing BATCH COUNT by 1 is carried out by adding 1000 which is thought of as 1.000. COUNT is stored as a whole number and therefore is incremented by adding 1. Nibbles A0-AC Hex of BAM1 are used to store a BCD 1000 which is designated REGTH01 when addressed as nibbles A0-A9 Hex and as REGONE1 when addressed as nibbles A3-AC Hex. Corresponding locations in BAM2 are designated REGONE2 and REGTH02.

If microcomputer 90 discovers a condition requiring that the meter be set to a faulted condition a fault code is first written into nibble 2D Hex of both BAMS so that subsequent examination of the BAM contents can tell the factory what happened. The particular hexadecimal codes for various fault conditions are as follows:

F Hex - An error is detected after writing a nibble to a BAM and reading and verifying the newly written nibble.

7 Hex - An arithmetic error is detected when adding a new character to the keyboard register according to the checked arithmetic algorithms to be described below.

5 6 Hex - A disagreement between the corresponding permanent registers in BAM1 and BAM2 is detected either before retrieving the register value or after updating the register value.

10 3 Hex - The print head leaves its home position before positioning of the print wheels is completed, indicating a problem with solenoid 85 or home sensor 300.

2 Hex - The print head does not leave its home position within 100 milliseconds of receiving the debounced clutch signal, indicating a problem with home sensing switch 300.

15 Nibble 2E Hex of both BAMs is initialized at the factory and is used to designate the maximum number of digits that can be entered into the keyboard register while meter 5 is in the "post office" mode. Nibble 2F Hex of both BAMs is initialized at the factory and describes the type of meter as follows. A "1" in Bit 1 indicates that the meter is a United
20 Parcel Service (UPS) meter which has no lockout when the descending register goes negative. A "1" in Bit 2 indicates that the meter is a fractional cents meter. A "1" in Bit 3 indicates that there are four rather than three stepping
25 motors. Nibble 5A of BAM1 is used to store a keyboard character counter in order to tell the keyboard register routine when the keyboard register is full. Bit 0 is always "1."

Background operations that have an effect on DR, AR, or TOTAL are carried out according to checked arithmetic
30 algorithms. There are three different arithmetic operations, the addition of postage in the "post office" mode, the decrementing of DR and corresponding incrementing of AR during a normal printing cycle, and a checking routine used during initialization, keyboard entry, and clearing the
35 keyboard.

The basic premise of all the checked arithmetic algorithms is that the descending register DR and the ascending register AR must sum to the control total. In keeping

with the fault tolerance aspect of the meter, there is one copy of each of these three registers in each BAM. During arithmetic operation, these registers are moved to the temporary registers in the BAMs described above. Prior to such a move, a verification is made that the contents of corresponding BAM registers are equal.

The contents of the temporary registers at the completion of various routines and tasks are shown in tabular form in Fig. 19. The particular sequences giving rise to these final states are now described.

A subroutine designated CHKTOT, carries out a number of the above verifications in three places - during initialization, during a keyboard clear, and while adding to the keyboard register in "mail room" mode. Subroutine CHKTOT checks both copies of the three permanent registers (DR, AR, TOTAL) to make sure that they are equal, moves them to temporary locations and checks that their temporary location values hold to the equation $TAR + TDR = TTOTAL$ (TEMP2). It would require an extremely unlikely occurrence of two errors exactly complimenting each other to have a wrong TAR, TDR, or TTOTAL. As will be seen below, the other checked arithmetic algorithms carry out manipulations on TDR1, TDR2, and/or TEMP1 and thereafter update the permanent registers in both BAMs with these values. In order to facilitate reconstruction should microcomputer 90 cease to function properly before all updated registers have been copied into their permanent location, the non-updated descending and ascending registers are maintained in TDR2 and TAR2 respectively. The operation of CHKTOT is as follows:

1. DR1 and DR2 are compared.
2. DR1 is moved to TDR1 and TDR2.
3. TOTAL1 and TOTAL2 are compared.
4. TOTAL1 is moved to TEMP1 and TEMP2.
5. AR1 and AR2 are compared.
6. AR1 is moved to TAR1, TAR2, and TEMP1.
7. TDR1 is added to the copy of AR1 in TEMP1 and stored in TEMP1.

8. TEMP1 and TEMP2, which should both equal TOTAL are compared.

Although the print task and keyboard task routines will be described in detail below, the checked arithmetic aspect of those tasks will be described at this point. As each character is entered, a keyboard register, designated KB, is updated to reflect the entire entry since the last keyboard clear. The basic algorithm for keyboard entry is that each character entry causes new temporary values of AR and DR to be calculated, being given by $AR + KB$ and $DR - KB$, respectively, and to be stored in TAR1 and TDR1, respectively. The updated temporary register values are checked to make sure that they sum to TOTAL. The particular sequences of steps for updating the temporary registers upon entry of a keyboard character is as follows.

1. The keyboard register KB is updated to reflect the total keyboard entry.
2. CHKTOT is called to get new copies of AR1 in TAR1 and TAR2 and of DR1 in TDR1 and TDR2.
3. KB is added to the copy of AR1 stored in TAR1, and the result is stored in TAR1.
4. KB is subtracted from the copy of DR1 stored in TDR1, and the result is stored in TDR1.
5. The updated TAR1 is moved to TEMP1.
6. The updated TDR1 is added to TEMP1 and the result stored back in TEMP1.
7. TEMP1 and TEMP2 (the latter having a checked copy of TOTAL) are compared to make sure that they are equal.

In a print cycle, updating of the permanent registers occurs after the print head has left home, or is expected to leave home as evidenced by either the LEFT HOME signal or the receipt of CLUTCH signal. Updating occurs as follows:

1. TDR1 is copied into DR1 and DR2.
2. TAR1 is copied to AR1 and AR2.
3. Batch and count arithmetic is done.
4. After returning to home, CHKTOT is called to check that the updated permanent registers are equal and hold to the required arithmetic constraint.

Additional checked arithmetic procedures are carried out when the meter is in the "post office" mode. When the meter is in the "post office" mode, the temporary registers are not updated during keyboard entry as described above in connection with the normal keyboard entry prior to a print cycle. Thus, prior to a change by the postal service employee, the state of the temporary registers is like that immediately following a call to CHKTOT. That is, the temporary registers have the current permanent values of the ascending and descending registers and the total.

As discussed above, special keys 40 and 42 allow the postal service employee to change the value of DR and TOTAL by adding or subtracting a corresponding amount (the keyboard register) from both. Thus, depression and release of clear batch key 42 causes the keyboard entry to be added while if clear KB key 40 is depressed while key 42 is also depressed, release of key 42 causes the keyboard entry to be subtracted. The basic arithmetic constraint is that the updated descending register ($DR \pm KB$) when added to AR must equal $TOTAL \pm KB$. Upon releasing clear batch key 42, the following steps take place.

1. KB is added/subtracted to/from TDR1 and the result stored in TDR1.
2. A check is made that DR will not be greater than \$99,999.99 or less than zero.
3. KB is added/subtracted to/from TEMP1 (which contains TOTAL).
4. TEMP1 is moved to TEMP2.
5. TDR1 and TAR1 are added and the result stored in TEMP1.
6. TEMP1 and TEMP2 are compared.
7. TDR1 is moved to DR1 and DR2.
8. TEMP1 (new total) is copied to TOTAL1 and TOTAL2.
9. CHKTOT is called to insure that the moves were done without error.

In view of the above description of the foreground routine and the checked arithmetic algorithms, the initial-

ization routine can be understood. The initialization routine is executed when MPCLR goes low in a power up cycle to permit operation of microcomputer 90, and performs those functions necessary to bring meter 5 into operating condition in a valid, known state. These functions may be summarized as follows:


1. Solenoid 85 is turned off (SON low) to inhibit printing unless specifically carried out under control of microcomputer 90.
2. The memory locations in RAM, the memory allocation of which is illustrated in Fig. 14, are set to zero.
3. The foreground registers are initialized with the foreground disabled so that the foreground routine will start triggering one-shots 113 and 114.
4. Timer PRCTR is set to provide an interval of 100 milliseconds, to allow the power to stabilize. Then the timer and its interrupt are enabled. After the 100 milliseconds the power loss interrupt in microcomputer 90 is enabled. By this time power is stable and SYSCLR is false and operation of the meter commences.
5. A check is made that the fault nibbles in BAMS 100 and 102 are zero. As discussed above, when meter 5 is set to a faulted condition, it should be impossible to bring the meter up in a power up cycle. Accordingly, detection of a non-zero value here indicates either a lack of ability to read a good zero from the fault nibbles, or a malfunction of the system reset circuitry or of the BAMS themselves. If a non-zero value is found, the meter is set to a faulted condition but no specific error code is written.
6. The information relating to meter type and maximum digit entry is recovered from the BAMS and written to registers in RAM for ready availability to the operating program.
7. The keyboard is cleared by executing a clear keyboard routine which in turn calls CHKTOT to check that the BAM contents are equal and hold to the arithmetic constraint.
8. REGONE1 and REGONE2, and REGTH01 and REGTH02 are checked to make sure that they are respectively equal.

9. The foreground is enabled for 100 milliseconds to update the switch registers.

10. The foreground is disabled and a check is made to see if the meter is on a base (\overline{MB} low). If the meter is
5 not on a base, the keyboard is cleared and the background dispatcher loop is entered where it is expected that the register display WAKEUP bit will be set.

11. Assuming the meter is on a base, the printer registers are set up and the stepper motors are stepped to
10 9999 without verification by sending out more step commands than are necessary. This provides for the possibility that the stepper motors are stiff or sloppy after a period of disuse. Once it is determined that the print wheels are at 9999, the meter is stepped to 0000 with verifying. The meter
15 continues to try until reaching 0000, and will loop to the beginning of step 11 above if it can't. The background dispatcher loop is then entered, with the register display and keyboard WAKEUP bits set so that the keyboard register and selected register will be displayed.

20 Figs 20a, 20b and 20c, taken together, form a flow chart of the print routine for controlling postage printing and monitoring the operation of the print mechanism. Security is maintained by requiring the various events in a print cycle to occur in a well-defined sequence, and within
25 predetermined time intervals. The meter is set to a fault (or soft fault) state if any of the relevant signals are inconsistent with the meter's being in a known and expected state. Broadly, the print routine positions stepper motors 45-48 (and their respective print wheels therewith) on
30 receipt of a TAPE or ENVELOPE signal. When positioning is completed, solenoid 85 is turned on, and the CLUTCH and/or LEFT HOME signal is awaited. When the LEFT HOME signal indicates that the print head has left the home position or the CLUTCH signal indicates that the clutch has been pulled,
35 AR and DR are updated from temporary locations in the BAMS and the batch and count registers are updated. Upon a return of the print head to its home position, the solenoid is then turned off. If only a single tape is to be printed the



keyboard register is cleared. Either way, new temporary values ($TAR = AR + KB$, and $TDR = DR - KB$) are calculated.

During execution of the print routine keyboard entry and receipt of I/O signals are blocked. If while
5 printing in response to an ENVELOPE signal, another ENVELOPE signal is received, the print routine does not end upon the print head's returning to the home position, but rather leaves the solenoid energized and calculates a new value of TAR and TDR. The print routine then leaves itself in a state
10 where it is waiting for the CLUTCH and/or LEFT HOME signal after as if it had just completed positioning.

Before discussing the operation of the print routine in detail, reference should be had to Fig. 16 which illustrates schematically a portion of the data base from
15 which the routine operates. Each stepper has two registers allocated to it, one designated STEPIN(X) and one designated STEPTK(X) where X is the number of the stepper. The most significant nibble of STEPIN carries the instruction indicating which of the four windings is to be energized. The least
20 significant nibble of the STEPIN contains a binary decimal code for the Final Position the particular stepper motor is to assume prior to printing. This is normally determined from the appropriate digit of the keyboard register. The least significant nibble of STEPTK contains the Next Position
25 to which the stepper is to be stepped, while the most significant nibble of STEPTK carries separate bits of information, indicating the direction to step, whether the present position (verified by reading the switches) is equal to Next Position, and also if equal to Last Position, whether a
30 second try is being made to reach the next position, and whether the next position expected is a half position.

A register designated PRTSTA carries separate bits or groups of bits indicating which stepper is currently being stepped, the current printer task as will be described in
35 detail below, whether the initialization routine is being executed, and the status of certain bits from REGSW (TAPE, ENVELOPE and REPEAT). Additionally, reference is made to those bits in R7FORG signifying the status of the MB, LEFT HOME, and CLUTCH signals.

The print routine is responsible for performing multiple functions at different times during a print cycle. In order to maintain a proper sequence, the routine performs tasks in a fixed order. At the completion of a given task, the routine updates the relevant bits in PRSTA so that subsequent entry into the routine will cause the next task in sequence to be carried out. The order of the tasks, and the numerical code corresponding to the different tasks are as follows:

10 Task 0 - This is the state at the initiation of a print cycle and indicates that no task is pending. During task 0, the routine builds up portions of the required data base.

15 Task 1 - The routine sends out the most recently executed commands (from the previous print cycle) in order to make sure that the positions of the steppers correspond to those that are known (i.e. stored in STEPIN). This is done to take account of the possibility that one or more of the steppers has been jarred away from its last position since the last positioning operation.

20 Task 2 - The routine verifies that the steppers are in fact at their last position. Then the keyboard digits are copied to STEPIN to signify the final positions to be achieved, and the appropriate directions of stepping are chosen.

25 Task 3 - The routine sends out commands at 1.6 millisecond intervals to cause the steppers to step to their final (keyboard) positions.

30 Task 4 - This task is carried out once the stepper positioning has been completed. The routine is waiting for a signal indicating that the clutch has been pulled and/or that the print head has left its home position. If the CLUTCH signal is received first, the routine sets a 100 millisecond timer, within whose interval the LEFT HOME signal from the home sensor must be received. Once the print head has left its home position or the clutch is pulled, the BAM registers AR and DR are updated to reflect the amount of postage being printed on this cycle and the BATCH and COUNT registers are updated.

Task 5 - This task checks that the CLUTCH signal has gone away and that the print head has returned to its home position. At this point, the task is set to 0.

5 Tasks 6 and 7 - These tasks are carried out rather than tasks 4 and 5, respectively, if a new envelope signal is received once positioning has been carried out. Upon reaching task 7, the task is set to 4 to prepare for another printing operation.

10 Referring to Fig. 20a, the sequence of the print routine may be understood. It should be recognized that during a print cycle, the print routine may be entered multiple times in response to the TAPE/ENVELOPE WAKEUP, the HOME/CLUTCH WAKEUP or the TIMER WAKEUP. The routine immediately checks whether the meter is in the "post office" mode
15 (logical branch 930), in which case the meter soft faults (block 932). Finding the meter to be in the "mail room" mode, the routine disables the foreground (block 935), checks that there are sufficient funds remaining (logical branch 937), and checks to determine whether all the steppers are at
20 their final positions (logical branch 940). Assuming the positioning has been completed, the routine carries out tasks 4-7 as will be discussed below.

Prior to carrying out tasks 0-3 which effect positioning of the print wheels, the routine checks to make sure
25 that the HOME/CLUTCH wakeup bit is not set (logical branch 942). If the HOME/CLUTCH WAKEUP is found to be set, the routine checks the cause (logical branch 945). An indication from the LEFT HOME signal that the print head has left its home position prior to the completion of positioning evi-
30 dences a situation in which meter security is compromised since solenoid 85 or home sensor 300 may be malfunctioning. This is responded to by causing the meter to fault with hexadecimal code 3 (block 947). The presence of an
35 unexpected clutch signal is generally caused by a base malfunction or improper paper handling and does not represent a threat to security. However operation must be suspended, and to this end the meter soft faults (block 950).

Assuming the HOME/CLUTCH WAKEUP is not present, a three way branch is made according to the current task (block 952). If the current task is 0, as is true on the first pass through the print routine during a particular print cycle, a
5 "PRINTER ONLY" bit is set to inhibit the keyboard and I/O, relevant switch data from REGSW is loaded into PRSTA, and the task is set to 1 (block 955). The PRINTER ONLY bit remains set for the duration of the print cycle, and causes the main program to avoid testing whether I/O or keyboard requests are
10 pending. From this point, the sequence of instructions followed is the same as if the task on entry was 1.

The basic constraints on the print positioning timing are dictated by the need to carry out the overall positioning within approximately 0.25 seconds as determined
15 by the time it takes an envelope moving down the feed path to reach a position under the print head. Every command to a stepper lasts 6.4 or 12.6 milliseconds (if not able to move to the next position within 6.4 milliseconds). In order to result in a more even power drain, stepper commands are
20 staggered at 1.6 millisecond intervals. The TIMER WAKEUP is turned off, a 1.6 millisecond timer is restarted and the initial command is sent (block 957). The initial command is in fact the last command that was sent to the particular stepper to make sure that the stepper is where it is thought
25 to be, any possible difference being due to mechanical vibration and the like.

Tasks 1-3 are shown more specifically in Fig. 20b. Still in task 1, the routine tests whether all four steppers have been serviced (logical branch 960). If not, PRSTA is
30 adjusted to prepare for the next stepper in sequence (block 962). The routine then returns to the main program after doing an initialization check which does the following: checks that in the initialization mode (logical branch 965) and that initialization is completed (logical branch 967).
35 If it is, the steppers are turned off and the TIMER, TAPE/ENVELOPE, and HOME/ CLUTCH WAKEUP bits are cleared (block 970) before returning to the main program.

The routine then loops so that on subsequent passes the initial command is sent to the remaining steppers (block 957), and when it is determined at logical branch 960 that all four steppers have been done, the task is set to 2 (block 5 972). The bits in PRSTA are set for the next stepper in the cycle (which at this point is the first one), and after the initialization check, the routine returns to the main program.

On the next pass through the print routine, the 10 task is 2. After branching at block 952, the routine turns off the TIMER WAKEUP and restarts the 1.6 millisecond timer. The routine tests whether the stepper is at its next position (logical branch 975), and if it has not reached its next position within the 6.4 milliseconds, a bit is set (block 15 978), assuming that this was not the second try as determined at logical branch 980. If it was a second try, then it is checked to see if doing initialization (logical branch 981). If yes, a jump is made out of the print routine to completely restart the initialization of the printer routine from stepping to '9999' (block 982). Else, a soft fault is done 20 (block 983).

Assuming the stepper has reached its next position, the routine branches according to whether task 2 or 3 is being executed (logical branch 984). If all four steppers 25 have been serviced (logical branch 985), the routine checks whether they have all reached their next position (logical branch 988). For task 2 the next position is the final position.

If all the steppers have reached their next position, the routine branches (according to whether task 2 or 30 task 3 is being executed (logical branch 990). On task 2 the digits of the keyboard register are copied to the least significant nibbles of the respective STEPIN registers (block 992). The routine then determines the direction of stepping 35 (block 995). The task is then set to 3 (block 1000) and the routine returns to the main program.

On a subsequent entry during task 3, the routine branches at logical branch 982 in order to check whether the



final position has been reached (logical branch 1002). If the final position has not been reached, the TIMER WAKEUP is turned off, the 1.6 millisecond timer is restarted, and the next command is sent out (block 1005). Once the final position has been reached, the routine tests whether all steppers have been serviced (branch 985), and branches as in task 2. Following logical branch 990, the timer is turned off, the solenoid is turned on and the task is set to 4 (block 1007) before doing the initialization check and returning to the main program.

On subsequent passes through the routine, it is ascertained at logical branch 940 that positioning has been completed, and tasks 4-7 are carried out. These tasks are shown specifically in Fig. 20c.

The routine first checks whether a TIMER WAKEUP has occurred (logical branch 1010). As will be discussed below, on subsequent tasks, this test may be true, but on task 4 it will not. The routine checks whether the TAPE/ENVELOPE WAKEUP bit is set (logical branch 1012) and if it has, the routine checks whether another envelope is on the way (logical branch 1015). If so, the task is set to 6 or 7, depending upon whether it was 4 or 5 (block 1017). The TAPE/ENVELOPE WAKEUP is turned off prior to a return to the main program (block 1020). Assuming that neither the TIMER WAKEUP nor the TAPE/ENVELOPE WAKEUP had been set, the only cause for entry into the print routine is that the HOME/CLUTCH WAKEUP bit had been set. This WAKEUP bit is turned off (block 1022) and a test is made whether the LEFT HOME signal indicates that the print head has left its home position (logical branch 1025). Assuming the print head has not left its home position, the routine checks whether the CLUTCH signal is present (logical branch 1027). If the task is 5 or 7, the routine returns to the main program at logical branch 1030. Otherwise, a 100 millisecond timing interval is set (block 1032). The purpose of this timing interval is to enforce the requirement that the LEFT HOME signal is received within 100 milliseconds of the CLUTCH signal. A failure of this to occur evidences a loss of home sensor 300, which loss

would leave the meter with no redundancy in the event of a failure in the base. This failure manifests itself by the foreground's setting the TIMER WAKEUP. This will be detected on subsequent entry at logical branch 1010, and is handled by
5 causing the meter to fault with hexadecimal code 2 (block 1033). After setting the 100 millisecond timer, which occurs on task 4 or 6, the ascending and descending registers (AR1, AR2, DR1 and DR2) and the BATCH and COUNT registers in the BAMS are updated and the task is set to 5 or 7 prior to a
10 return to the main program (block 1035).

If, at logical branch 1025, it is ascertained that the print head has left home, the timer is turned off (block 1037). On task 5 or 7 (as determined at logical branch 1038) the routine returns to the main program. Otherwise, the BAM
15 and task updates (block 1035) are made before returning to the main program.

If it is determined at logical branches 1025 and 1027 that the CLUTCH signal and the LEFT HOME signal are both absent, the routine returns to the main program on tasks 4 or
20 6 (logical branch 1040). If the task is 5 or 7, this state indicates that the print head has returned to its home position. The routine tests whether the task is 7 (logical branch 1045), and turns off the steppers and solenoid (block 1047), if the task is 5, while they are left on if the task
25 is 7. If task 5, the task is set to 0 (block 1052) while if the task is 7, the task is set to 4 (block 1055). New updated values of the registers are stored in the temporary registers in the BAMS (block 1057). If the task is 5, a check is made whether a single tape was to be printed
30 (logical branch 1060). If a single tape was to be printed, the keyboard register is cleared and the routine calls the subroutine CHKTOT (block 1062) to make sure that the new registers adhere to the basic arithmetic constraint. Where envelopes or multiple tapes are to be printed, the temporary
35 registers and keyboard register are set up to so that a subsequent print cycle can occur without requiring a new keyboard entry. A single tape (not repeat mode), on the other hand, once printed, causes the keyboard register to be cleared.

Prior to returning to the main program, the available funds are checked (logical branch 1065) and if it is found that insufficient funds remain to print another tape or envelope of the keyboard register value, "Add \$" light 22 is
5 lit, the steppers and solenoid are turned off, and the task is set to 0 (block 1067). If the task has been set to 4, indicating that a new print cycle is to occur, the program returns to the main program while if the task is 0 the PRINTER ONLY bit is turned off to enable the keyboard and I/O
10 (block 1070).

Figs. 21a and 21b, taken together, form a flow chart of the keyboard routine. Generally, the keyboard routine discovers what keyboard key or keys have been pressed and takes action accordingly, as for example by updating the
15 keyboard register or various BAM registers. This routine is entered when the background dispatcher loop detects the KEYBOARD WAKEUP bit to have been set, which as discussed above occurs when the foreground routine detects a change to REG89 or REG07. In addition to keyboard key status, REG89
20 contains the status of switches 27 (fractional cents entry) and 30 (POMODE). The flow through this routine is controlled by the least significant nibble of KEYSTA, designated CC for current character and by a 1-bit flag, designated F0. Usually CC is set equal to the current keyboard switch being
25 pressed, but it may assume values outside the range 0-9 to indicate special circumstances. In particular, if no keys are depressed, CC = 10; if keys "7", "8", and "9" are all pressed to signify that all the display segments are to be energized, CC = 12; and if the keyboard register KB is full,
30 CC = 14. As discussed above, in the "mail room" mode, four characters may be entered before the keyboard register KB is considered full (three if the meter 5 only contains three steppers). In the "post office" mode, the entry capability expands to the number of characters in the maximum permissible value (e.g. seven characters if the maximum value
35 allowed is \$99,999.99). The determination whether KB is full is on the basis of the keyboard register character counter KBCC which is the tenth nibble of KB.

Before describing the precise operation and sequence of the keyboard routine, it is helpful to consider the various ways in which exit from the keyboard routine occurs. Generally, a return to the main program (background dispatcher loop) occurs via one of four exit points 1100, 1102, 1105, and 1107, designated A, B, C, and D, respectively. Upon branching to exit point A, the routine tests whether the printer is still busy (logical branch 1110), returning to the main program if it is and turning off the I/O and KEYBOARD WAKEUP bits and the PRINTER ONLY bit if it is not, prior to returning to the main program (block 1112). Upon branching to exit point B, the routine sets CC = 10 (block 1115) and returns via exit point A. Upon branching to exit point C, the routine clears the keyboard register and sets up a register request (block 1117), displays the keyboard register if the meter is in the "mail room" mode (block 1120), and returns via exit point B. Upon a branch to exit point D, the routine lights "Add \$" lamp 22 clears the keyboard register, sets CC = 14 (block 1122), and returns via exit point A.

Upon entry into the keyboard routine, the foreground is immediately disabled and the routine initialized (block 1125). For definiteness, consider first the case where a single numeric key has been pressed. Prior to entry into the routine, CC will have been initialized to 10 (or left set at 10 from the last time that all keys had been released). The routine tests whether CC = 12 (block 1127), and upon finding it not equal to 12, sets flag F0 = 1 only if CC = 14 (block 1130). On this entry, F0 will not be set equal to 1. The routine then tests whether keys "7", "8", and "9" are all depressed (logical branch 1132), and upon finding such not to be the case, the routine undertakes to discover which key has been depressed (software vector 1135). This test is made according to a specific hierarchy wherein the keys are tested in the order "7", "8", "9", "4", "5", "6", "1", "2", "3", CLEAR BATCH, "0", and CLEAR KB. This particular order is appropriate for a standard calculator configured keyboard and in some measure avoids the problem



wherein a right handed person, in addition to the intended key, depresses the key immediately to the right or below. Assuming a numeric key to have been discovered as the first key to be depressed, the routine tests whether the keyboard register is full, i.e. whether F0 = 1 (logical branch 1137).

5 If the keyboard register is full, the routine returns to the main program via exit point A. If it is not full, a test is made whether CC = 10 (logical branch 1140) and returns via exit point A if CC is not equal to 10 since
10 this would indicate another key to still be pressed. If CC = 10, the routine sets CC equal to the numeric character found (block 1142). The routine then increments KBCC and updates the keyboard register by left shifting the current contents and entering the newly discovered character in the rightmost
15 position (least significant) (block 1145). The precise nibble which is considered the rightmost or least significant position is nibble 0 if the meter is in the "mail room" mode, and is a fractional cents meter, and selector switch 27 is set for fractional cents entry. Otherwise, it is nibble 1 of
20 the keyboard register. If with the additional character, the keyboard register is full, the routine sets CC = 14 (block 1147), and then sets up a register request (block 1150). The routine then tests whether the meter is in the post office mode (logical branch 1152) and returns via exit point B if it
25 is. Otherwise, the temporary registers TDR and TAR in the BAMs are updated as described above (block 1155). The checked arithmetic algorithm is carried out to make sure that the updated TAR and TDR sum to the control total (logical branch 1157), and if not, the meter faults with a hexadecimal
30 code 7 written into the BAM location reserved for that purpose (block 1160). Under normal (non-fault) circumstances the routine then displays the new keyboard register (block 1162), tests whether the updated decending register has become negative (logical branch 1165), and if not, returns
35 via exit point A. If TDR has become negative, the routine returns via exit point D.

Assume that keys "7", "8", and "9" are all depressed and that the current pass through the routine is in



response to the last one of these keys to have been depressed. At logical branch 1132, the routine branches differently from the above described sequence, sets up the segment codes so that all segments in the displays will be illuminated, and clears the keyboard register (block 1167).
5 Then the routine sets CC = 12 (block 1170) prior to a return via exit point A. The next pass through the keyboard routine finds CC = 12 at logical branch 1127, whereupon the routine tests whether all of keys "7", "8", and "9" have been
10 released (logical branch 1172). If not all of these three keys have been released, the routine returns via exit point A, but if all of the keys have been released, the routine returns via exit point C in order to clear the keyboard register, turn off the display of all segments, and set CC =
15 10.

If, at software vector 1135, it is determined that CLEAR KB is the only key to be depressed, the routine exits via exit point C. If it is found that no key is currently being depressed, the routine tests whether the entry into the
20 routine was due to a new change into the "post office" mode (logical branch 1175) and returns via exit point C if it was. Otherwise, the routine returns via exit point B if CC is found equal to 12 or 14, and by exit point A otherwise (logical branch 1177).


25 It will be recalled that action in response to the CLEAR BATCH key occurs when the key is released, and may require the CLEAR KB key to have been depressed in the interim. In the event that the CLEAR BATCH key is discovered to have been depressed, the routine initially sets F0 = 0
30 (block 1180) and enables the foreground (block 1182) in preparation for waiting until the key is released. The routine then executes a wait loop, 1185, during which it tests whether F0 has been set to 1 (logical branch 1187). Initially, F0 will be 0 and the routine sets F0 = 1 when and
35 if it finds the CLEAR KB switch to have been depressed (block 1190)). The routine then tests whether the CLEAR BATCH key has been released (logical branch 1192), and if not, loops back to logical branch 1187. When the clear batch key is

released, as discovered at logical branch 1192, the foreground is again disabled (block 1195).

The routine tests whether the meter is in the "post office" mode (logical branch 1197), and if so, adjusts
5 temporary register TDR up or down by the keyboard register contents, depending on whether the CLEAR KB key had been depressed during the time that the CLEAR BATCH key had been pressed (block 1200). The routine then tests whether the
10 descending register value would be in the permissible range (logical branch 1202). If not, the routine returns via exit point D in order to display the "Add \$" lamp. If the descending register value is a permissible one, the routine adjusts the control total temporary register in the same direction as it had adjusted the descending register and
15 updates permanent registers DR and TOTAL (block 1205) prior to returning via exit point C. If at logical branch 1197, the routine finds that the meter is in the "mail room" mode, the routine tests whether $F0 = 1$ (logical branch 1207) to determine whether the CLEAR KB key had been depressed during
20 the time that the CLEAR BATCH key was depressed. If so, the routine clears the keyboard and batch registers (block 1210) prior to a return via exit point C. Otherwise, the routine returns via exit point B and in effect ignores depression and release of the CLEAR BATCH key.

25 The register display routine is entered in response to a REGISTER DISPLAY WAKEUP, and is responsible for determining which register is to be displayed and displaying it. The routine determines which display to display the information on, and takes care of decimal point placement and blanking leading zeros. The register display routine is straight-
30 forward and will not be described in further detail.

The I/O routine is entered in response to an I/O WAKEUP and has the function of receiving signals from outside the meter in order to carry out those functions that would be
35 carried out in response to various combinations of keyboard key depressions and selector switch positions. The I/O routine will not be described in detail.



In summary, it can be seen that the present invention provides a microcomputerized postage meter having a high degree of security and fault tolerance so that critical register data is preserved under almost any conceivable failure condition. Once a failure has occurred, the meter recognizes its own lack of redundancy, and hence susceptibility to losing the data, and responds to this condition by activating internal circuitry for disabling the meter and preventing further operation until the meter is reset at the factory. Although disabled, the data in the registers is accessible for possible diagnostic purposes.

While the above description provides a full and complete disclosure of the preferred embodiment of the invention, various modifications, alternate constructions, and equivalents may be employed without departing from the true spirit and scope of the invention. For example, while the circuitry for inhibiting meter functioning upon detection of a fault condition includes a specific type of flip-flop, other flip-flop types or settable-resettable circuit elements could be adapted for use with the present invention. Similarly, while the use of stepper motors and mechanical verification contacts represents a preferred and relatively economical way to accomplish print wheel setting, other actuating mechanisms will be readily apparent to those of ordinary skill in the art. Also, the particular keyboard sequences and responses, while representative of appropriate data management, can be varied so long as such variations are carried out in a consistent manner. Accordingly, the above description and illustration should not be construed as limiting the scope of the invention, which is defined by the appended claims.

A computer program for the invention is shown by way of example in appended Appendix 1.

IN THE CLAIMS:

1. In a microcomputerized postage meter having a microcomputer programmed for supervising the printing operation and for maintaining and verifying accounting information, a postage printing mechanism coupled to the micro-
5 computer for printing postage in response to instructions from the microcomputer, and input means coupled to the microcomputer for communicating data to the microcomputer; the improvement comprising:
- 10 first and second independent non-volatile memory means coupled to the microcomputer, each non-volatile memory means including self-contained power supply means and a plurality of memory locations corresponding to accounting information;
- 15 the programmed microcomputer including means for storing an item of accounting information in corresponding locations of the first and second non-volatile memory means, for retrieving the contents of the corresponding locations, and for comparing the contents of the corresponding locations; and
- 20 means coupled to the microcomputer for deactivating the meter in response to a disagreement between the contents of the corresponding locations.

2. In a microcomputerized postage meter having a microcomputer programmed for supervising the printing operation and for maintaining and verifying accounting information, a postage printing mechanism coupled to the micro-
5 computer for printing postage in response to instructions from the microcomputer, and input means coupled to the microcomputer for communicating data to the microcomputer; the improvement comprising:

10 non-volatile memory means coupled to the microcomputer including self-contained power supply means and a memory location corresponding to accounting information;

the programmed microcomputer including means for storing an item of accounting information in a location of the non-volatile memory means, and for retrieving the
15 contents of the location;

means for retrieving the contents of the location while the microcomputer is in an inactive state with respect to the non-volatile memory means.

3. The invention of claim 2 wherein the inactive state is an unpowered state.

4. The invention of claim 2 wherein the inactive state is a state wherein the microcomputer is incapable of accessing the non-volatile memory means.

5. In a microcomputerized postage meter having a microcomputer, a postage printing mechanism coupled to the microcomputer for printing postage in response to instructions from the microcomputer, and input means coupled to the microcomputer for communicating data to the microcomputer;
5 the improvement comprising:

first and second independent non-volatile memory means coupled to the microcomputer, each non-volatile memory means including self-contained power supply means and a
10 plurality of memory locations;

the programmed microcomputer having means for storing a set of accounting information items, the values of which items have an internal relationship to one another irrespective of their individual values, in corresponding
15 pluralities of locations in the first and second non-volatile memory means, for retrieving the contents of the corresponding pluralities of locations, and for comparing the contents of the corresponding pluralities of locations; and

means coupled to the microcomputer for deactivating
20 the meter in response to a disagreement between the contents of the corresponding pluralities of locations;

such that the contents of the corresponding pluralities of locations permit an operator to determine which of the two disagreeing sets of values is correct by checking
25 which of the two sets of values satisfies the internal relationship.

6. In a microcomputerized postage meter having a microcomputer operable according to a program for supervising the printing operation and for maintaining and verifying accounting information, postage printing means coupled to the microcomputer for printing postage in response to instructions from the microcomputer, and input means coupled to the microcomputer for communicating data to the microcomputer; the improvement comprising:
- first and second independent non-volatile memory means coupled to the microcomputer;
 - monostable multivibrator means having an input and an output;
 - the multivibrator output capable of assuming first and second distinct logic levels;
 - the multivibrator means being operable to maintain the multivibrator output at the first level for a predetermined time in response to a signal;
 - means coupled to the multivibrator output for disabling the meter in response to the second logic level on the multivibrator output; and
 - enable signal generating means coupled to the microcomputer and to the multivibrator input for repeatedly generating an electrical signal at intervals less than said predetermined time during operation of the meter;
 - the programmed microcomputer having means for suppressing the enable signal generating means in response to the detection of a failure condition, and means for storing in the first and second non-volatile memory means a code representative of the type of failure detected such that the detection of a failure causes the multivibrator output to assume the second logic level, wherein the meter is disabled.

7. In a microcomputerized postage meter having a microcomputer, and input means coupled to the microcomputer for generating signals representative of the value of postage to be printed, the improvement comprising:


5 a print element having a plurality of serially disposed positions, each position representative of a value of postage to be printed;

stepping motor means coupled to the print element, the stepping motor means having a plurality of positions
10 corresponding to the plurality of print element positions;

motor driving means coupled to the microcomputer for driving the motor means from a first of the plurality of positions to a second serially adjacent one of the plurality of positions; and

15 position indicator means coupled to the microcomputer for generating and communicating to the microcomputer an electrical signal representative of the position of the stepping motor;

the programmed microcomputer having means for
20 activating the motor driving means for a fixed time interval, means for determining whether the stepping motor means has moved from the first position to the second position within the fixed time interval, and means for suspending operation of the meter in response to a determination that the stepping
25 motor means has not moved from the first position to the second position within the fixed time interval.



8. In a microcomputerized postage meter having a microcomputer, a postage printing mechanism coupled to the microcomputer for printing postage in response to instructions from the microcomputer, and input means coupled to the microcomputer for communicating data to the microcomputer;
5 the improvement comprising:

a non-volatile memory unit including first and second independent non-volatile memory means coupled to the microcomputer, each nonvolatile memory means including self-
10 contained power supply means and a plurality of memory locations;

the programmed microcomputer having means for storing a set of accounting information items, the values of which items have an internal relationship to one another
15 irrespective of their individual values, in corresponding pluralities of locations in the first and second nonvolatile memory means, means for storing the set of accounting information items in a third plurality of locations in the non-volatile memory unit, means for changing the values of at
20 least one of the items stored in the third plurality of locations in response to a signal from the input means, the changed values maintaining the same internal relationship, and means for copying the changed values into the first and second pluralities of locations when the postage printing
25 mechanism is activated in accordance with the signals from the input means.

9. In a microcomputerized postage meter having a microcomputer, a postage printing mechanism coupled to the microcomputer for printing postage in response to instructions from the microcomputer, and input means coupled to the microcomputer for communicating data to the microcomputer, the microcomputer and the printing mechanism being located in a secure housing, the improvement comprising:
- non-volatile memory means coupled to the microcomputer, including self-contained power supply means and a plurality of memory locations for storing a set of accounting information items;
 - bistable multivibrator means including self-contained power supply means and having an output capable of assuming first and second distinct logic levels in response to first and second respective states of the bistable multivibrator means, the bistable multivibrator means, once in the second state, being resettable to the first state only by physical access to the interior of the secure housing;
 - means coupled to the microcomputer for causing the bistable multivibrator means to assume its second state in response to the detection of a failure condition;
 - means responsive to the logic level of the bistable multivibrator output for generating a system clear signal in response to the appearance of the second logic level on the bistable multivibrator means output;
 - power surveillance means for generating a power loss signal in response to a low power condition;
 - the means for generating the system clear signal being further responsive to the power loss signal for generating the system clear signal;
 - means responsive to the system clear signal for preventing data transmission to the non-volatile memory means; and
 - means responsive to the system clear signal for preventing activation of the printing mechanism.

10. The invention of claim 9 also comprising:
- means responsive to the appearance of the second logic level on the bistable multivibrator output for generating a microcomputer clear signal;
 - 5 the means for generating the microcomputer clear signal being further responsive to the power loss signal for generating the microcomputer clear signal; and
 - means responsive to the microcomputer clear signal for disabling the microcomputer.

11. The invention of claim 1 or 5 wherein the microcomputer and the printing mechanism are located in a secure housing, and wherein the means for deactivating the meter comprises:

- 5 bistable multivibrator means including self-contained power supply means and having an output capable of assuming first and second distinct logic levels in response to first and second respective states of the bistable multivibrator means, the bistable multivibrator means, once in the
10 second state, being resettable to the first state only by physical access to the interior of the secure housing;
 means coupled to the microcomputer for causing the bistable multivibrator means to assume its second state in response to the detection of a failure condition;
15 means responsive to the logic level of the bistable multivibrator output for generating a system clear signal in response to the appearance of the second logic level on the bistable multivibrator means output;
 means responsive to the system clear signal for
20 preventing data transmission between the microcomputer and the non-volatile memory means;
 means responsive to the system clear signal for preventing activation of the printing mechanism;
 means responsive to the appearance of the second
25 logic level on the bistable multivibrator output for generating a microcomputer clear signal; and
 means responsive to the microcomputer clear signal for disabling the microcomputer.

12. The invention of claim 11 also comprising:
power surveillance means for generating a power
loss signal in response to a low power condition;
the means for generating the system clear signal
5 being further responsive to the power loss signal for gener-
ating the system clear signal; and
the means for generating the microcomputer clear
signal being further responsive to the power loss signal for
generating the microcomputer clear signal.

13. The invention of claim 11 wherein the means
for preventing data transmission between the microcomputer
and the non-volatile memory means permits data to be read
from the non-volatile memory means independent of the
5 microcomputer.

LOC	OBJ	LINE	SOURCE STATEMENT
0017		55 METER EQU 017H	; ADDR OF METER TYPE INFO. BIT 7 = 1 SAYS 4 STEPPER.
		56	; METER. = 0 SAYS 3: BIT 6 IS FEEDS BIT; BIT 5 IS
		57	; 1 IF UPS METER, LSN TELLS HOW MUCH
		58	; P. 0 CAN ENTER INTO DE
0033		59 PRCF EQU 036H	; PRINT WAIT COUNTER IF HE 0 IS COUNTED DOWN BY FOREGROUND
		60	; ARE WHEN GOES TO ZERO IS REPORTED TO MAIN ROUTINE
		61	; THEN BASTA
0075		62 PPTSTA EQU 035H	; ADDR OF CURP STATUS OF PRINTER ROUTINE.
		63	; MSE SAYS INITIALIZATION. NEXT 3 BITS TELL
		64	; WHAT PHASE OF PRINTING. BITS 212 TELL IF
		65	; TAPE OF ENV. ARE REPEAT. ARE BITS 011
		66	; TELL WHICH STEPPER TO STEP IF NECESSARY.
0015		67 IOSTP EQU 016H	; ADDR OF CURP STATUS OF I/O ROUTINE
0027		68 IOPTR EQU 037H	; ADDR OF PTR USED BY I/O TO READ OF WRITE BAK
0025		69 SAVADR EQU 029H	; ADDR WHERE THE TWO INTERRUPT ROUTINES STORE THE
		70	; BACKGROUND'S ADDRESS
0002		71 BASTA EQU 002H	; ADDR (R2) OF BACKGROUND STATUS REG
0035		72 STEPIN EQU 009H	; ADDR OF 4 INDEXED BYTES GIVING INFO ABOUT
		73	; THE STEPPERS LSN = FINI DESIRED POSITION. MSN =
		74	; PRESENT COMPARE TO THE STEPPER
0030		75 STEPTK EQU 03CH	; ADDR OF 4 INDEXED BYTES GIVING FURTHER INFO
		76	; ABOUT THE STEPPERS MSN=STATUS INFO FOR EACH STEPPER.
		77	; LSN = NEXT POSITION CODE EXPECTED
0073		78 STEPTK2 EQU 03CH	; ADDR OF STEPTK(2)
		79	;
		80	; THE FOLLOWING DEFINITIONS ARE TO SET AND CLEAR BITS IN BASTA
		81	;
0003		82 KEYTSK EQU 002H	; FOR ORL A.# TO SET KEYSK TASK TO ONE
0021		83 INVEI EQU 001H	; FOR ORL A.# TO INITIAT I/O AND KE TASKS
0010		84 SWTSK EQU 010H	; FOR ORL A.# TO SET REG SWITCH TASK TO ONE
FFF1		85 NSWTSK EQU NOT SWTSK	; FOR AND A.# FOR CLEAR ABOVE TASK
0020		86 PTITSK EQU 020H	; FOR ORL A.# TO SET PRINTER TIMED TASK TO ONE
FFF0		87 NPTITSK EQU NOT PTITSK	; FOR AND A.# FOR CLEAR ABOVE TASK
0034		88 IOTSK EQU 004H	; FOR ORL A.# TO SET I/O TASK TO ONE
FFF6		89 NIOTSK EQU NOT IOTSK	; FOR AND A.# TO CLEAR ABOVE TASK
FFF2		90 NKEYIOP EQU NIOTSK AND NOT KEYTSK AND NOT INVEI	; FOR AND A.# AT
		91	; END OF KEYBOARD TASK WHEN NOT PRINTING
0000		92 PFTSK EQU 006H	; SET UPON FINDING POWER LOSS COMING TRUE
0002		93 PHOCL EQU 002H	; SET UPON FINDING CHANGE IN HOME OR CLUTCH
FFF1		94 NPHOCL EQU NOT PHOCL	; USED TO TURN OFF PHOCL
0040		95 PTREN EQU 040H	; SET UPON FINDING CHANGE IN TAPE OR ENVELOPE
FFF0		96 NPTREN EQU NOT PTREN	; FOR TURNING OFF TAPE/ENV MAKEUP
FFF0		97 NPTREN AND NPHOCL	; USED TO TURN OFF PTREN & NPHOCL
FFF5		98 NPTT1 EQU NPTREN AND NPTTSK	; USED TO TURN OFF PTREN & OR PTITSK
FFF3		99 NKB1OT EQU NOT KEYTSK AND NIOTSK	; USED TO TURN OFF THESE 2 BITS
		100	;
		101	; FOLLOWING DEFINITIONS ARE FOR THE VARIOUS BAK REGISTERS
		102	;
0000		103 BAREG EQU 000H	; DECREMENTING REG(A)
0010		104 BAREG EQU 010H	; ASCENDING REG(B)
0020		105 BTOTAL EQU 020H	; CONTROL TOTAL (C)
0025		106 BCFEY EQU 2CH	; WHERE ALL 4'S STORED TO VERIFY CORRECT
		107	; RECORDING OF BAK BY MAT
0021		108 BFAULT EQU 020H	; WHERE FAULT INFO IS STORED
0025		109 BMS20 EQU 025H	; WHEN MAY P. 0 MAKE CODE IS STORED

LOC	OBJ	LINE	SOURCE STATEMENT
0020		110 BBTYP EQU 02FH ;REPE BITS ABOUT THE METER TYPE ARE STORED	
0030		111 BBATL EQU 030H ;BATCH TOTAL (0)	
0040		112 BBACT EQU 040H ;BATCH COUNT (0)	
0050		113 BACOUNT EQU 050H ;COUNT OF PRINT CYCLES	
0060		114 BKBREG EQU 060H ;KEYBOARD REGISTER (0) (RAM 1 ONLY)	
0065		115 CKBCHA EQU 06AH ;ADDR OF 1 OF CHRS--IN BKBREG(R)	
0070		116 BTDPREG EQU 070H ;TEMP DREG(0) (RAM 1 ONLY)	
0078		117 BTDPPE EQU 078H ;LOCATION USED BY KETLG ROUTINE TO SEE IF	
		118 ;NEW VALUE OF DP WOULD BE TOO LARGE OR SMALL	
0080		119 BTAREG EQU 080H ;TEMP AREG(0) (RAM 1 ONLY)	
		120 ;TEMP AREG(0) (RAM 2) USED TO STORE TEMP COPY OF TOTAL	
0090		121 BTEMP EQU 090H ;TEMPORARY REGISTERS(0) (BOTH RAMS) USED TO DO	
		122 ;RXRS AND SURS W/O MODIFYING OTHER REGISTERS	
		123 ;AFTER WHICH CHECKS ON EQUIVALENCY ARE DONE	
0083		124 BRGONE EQU 0A3H ;REGISTER WITH VALUE OF 1 FOR INCREMENTING COUNT	
0085		125 BRGTHD EQU 0A2H ;REGISTER WITH VALUE OF 1 FOR INCP BATCH CT	
0088		126 BTDDR EQU 0E0H ;START OF TEMP REGISTERS FOR LAST CHANGE SAVE	
		127 ;	
		128 ;THE FOLLOWING DEFINITIONS ARE FOR MOND P5.A	
		129 ;TO SELECT THE PROPER 74150 DEMULTIPLEXER INPUTS TO TEST ON T1	
		130 ;	
0085		131 DMPCLU EQU 0A2H ;DEMPX ADDR FOR CLUTCH	
0088		132 DMPKIN EQU 0A0H ;DEMPX ADDR FOR KEYBOARD INPUTS	
0091		133 DMPKIN EQU 0A1H ;DEMPX ADDR FOR REG SELECT SW INPUT	
0095		134 DMPTRF EQU 0A5H ;DEMPX ADDR FOR TAPE SIGNAL INPUT	
		135 ;	
		136 ;	
		137 ;THE FOLLOWING DEFINITIONS ARE FOR THE PRINT ROUTINE	
		138 ;	
0088		139 STPM1 EQU 088H ;USED TO TELL NEXT EXPECTED POSITION IS A 1/2	
		140 ;POSITION (RETURNS -1)	
FF7F		141 NSTPM1 EQU NOT STPM1 ;USED TO NEGATE THE ABOVE	
0028		142 STPDON EQU 020H ;TELLS STEPPING MOTOR MOVED TO A FULL POSITION	
0058		143 POSDON EQU 060H ;STEPPER MOTOR POSITIONED TO FINAL POSITION	
FF9F		144 NDOKEP EQU NOT STPDON AND NOT POSDON ;STEPPER STARTING TO NEW POS	
0084		145 STINTV EQU 0A4H ;# OF TIME INTERVALS FOR EACH 1/4 PRINT CYCLE	
		146 ;400 MICRO SECS * 5 SVCS EACH STEP TAKES 2 MILLISECS	
0048		147 STPTRY EQU 040H ;SET TO INDICATE ALREADY TRIED TO POSITION 1	
		148 ;PRINT CYCLE (ABOVE TIME).	
		149 ;	
		150 ;THE FOLLOWING DEFINITIONS ARE FOR THE I/O ROUTINE	
		151 ;	
0081		152 IOPTR EQU 001H ;COMMAND TO READY NEW PTR FROM UP1	
0082		153 IOAKS EQU 002H ;COMMAND TO READ NEW CHR FOR KEYBOARD REG	
0083		154 IORDE EQU 003H ;COMMAND TO READ RAM USING IOPTR	
0084		155 IOAREX EQU 004H ;COMMAND TO WRITE TO RAM USING IOPTR	
0089		156 ENAIO EQU 009H ;PORT 6 DATA TO ENABLE I/O (EIO)	
		157 ;	
		158 ;*****	
		159 ;	
		160 ;	
		161 ; TIMINT -- S/R FOR TIMER INTERRUPT. THOUGHT OF AS A FOREGROUND TASK	
		162 ; , THIS DISPLAYS THE TWO DISPLAYS, DEBOUNCES THE KEYBOARD	
		163 ; SWITCHES, AS WELL AS THE REGISTER SELECT SWITCHES.	
		164 ; THE SIGNALS:TAPE, ENVELOPE AND REPEAT, READS THE SWITCHES	

LOC	TEXT	LOC	TEXT	LOC	TEXT
165	000 P O. MOVE, AND THE INTERRUPTS (1/0) IT ALSO	165	000 P O. MOVE, AND THE INTERRUPTS (1/0) IT ALSO	165	000 P O. MOVE, AND THE INTERRUPTS (1/0) IT ALSO
166	(RES COUNTING FOR THE PRINTER ROUTINE IF NECESSARY. IT	166	(RES COUNTING FOR THE PRINTER ROUTINE IF NECESSARY. IT	166	(RES COUNTING FOR THE PRINTER ROUTINE IF NECESSARY. IT
167	(COORDINATES WITH THE BACKGROUND TASKS THRU BRKSTA (R2), WHICH	167	(COORDINATES WITH THE BACKGROUND TASKS THRU BRKSTA (R2), WHICH	167	(COORDINATES WITH THE BACKGROUND TASKS THRU BRKSTA (R2), WHICH
168	THE BACKGROUND TASK MAIN LIES TO START VARIOUS TASKS. IT	168	THE BACKGROUND TASK MAIN LIES TO START VARIOUS TASKS. IT	168	THE BACKGROUND TASK MAIN LIES TO START VARIOUS TASKS. IT
169	ALSO SETS UP VARIOUS REGISTERS FOR THE BACKGROUND TO READ	169	ALSO SETS UP VARIOUS REGISTERS FOR THE BACKGROUND TO READ	169	ALSO SETS UP VARIOUS REGISTERS FOR THE BACKGROUND TO READ
170	TO TELL WHAT IS THE PRESENT STATE OF THE OUTSIDE WORLD	170	TO TELL WHAT IS THE PRESENT STATE OF THE OUTSIDE WORLD	170	TO TELL WHAT IS THE PRESENT STATE OF THE OUTSIDE WORLD
171	SO WHILE THE CURRENT REG. R1 IS USED FOR OTHER DUTIES.	171	SO WHILE THE CURRENT REG. R1 IS USED FOR OTHER DUTIES.	171	SO WHILE THE CURRENT REG. R1 IS USED FOR OTHER DUTIES.
172	R1 IS USED THEREFORE ONLY ROUTINES WHICH DISABLE TIMINT	172	R1 IS USED THEREFORE ONLY ROUTINES WHICH DISABLE TIMINT	172	R1 IS USED THEREFORE ONLY ROUTINES WHICH DISABLE TIMINT
173	CAN USE R1 TIMINT TAKES 115 MICROSECS EVERY 400 WHEN	173	CAN USE R1 TIMINT TAKES 115 MICROSECS EVERY 400 WHEN	173	CAN USE R1 TIMINT TAKES 115 MICROSECS EVERY 400 WHEN
174	DISABLED. HENCE ALL BACKGROUND TASKS TAKE 400/285 LONGER	174	DISABLED. HENCE ALL BACKGROUND TASKS TAKE 400/285 LONGER	174	DISABLED. HENCE ALL BACKGROUND TASKS TAKE 400/285 LONGER
175	THAN ACTUAL INTERRUPTION TIME	175	THAN ACTUAL INTERRUPTION TIME	175	THAN ACTUAL INTERRUPTION TIME
176		176		176	
177		177		177	
178	*****	178	*****	178	*****
179		179		179	
180	THIS IS START OF FRIEDEN ELECTRONIC POSTAL METER MICROCODE	180	THIS IS START OF FRIEDEN ELECTRONIC POSTAL METER MICROCODE	180	THIS IS START OF FRIEDEN ELECTRONIC POSTAL METER MICROCODE
181		181		181	
182	0000 0000 0000 0000	182	0000 0000 0000 0000	182	0000 0000 0000 0000
183	0000 0000 0000 0000	183	0000 0000 0000 0000	183	0000 0000 0000 0000
184	0000 0000 0000 0000	184	0000 0000 0000 0000	184	0000 0000 0000 0000
185	0000 0000 0000 0000	185	0000 0000 0000 0000	185	0000 0000 0000 0000
186	0000 0000 0000 0000	186	0000 0000 0000 0000	186	0000 0000 0000 0000
187	0000 0000 0000 0000	187	0000 0000 0000 0000	187	0000 0000 0000 0000
188	0000 0000 0000 0000	188	0000 0000 0000 0000	188	0000 0000 0000 0000
189	0000 0000 0000 0000	189	0000 0000 0000 0000	189	0000 0000 0000 0000
190	0000 0000 0000 0000	190	0000 0000 0000 0000	190	0000 0000 0000 0000
191	0000 0000 0000 0000	191	0000 0000 0000 0000	191	0000 0000 0000 0000
192	0000 0000 0000 0000	192	0000 0000 0000 0000	192	0000 0000 0000 0000
193	0000 0000 0000 0000	193	0000 0000 0000 0000	193	0000 0000 0000 0000
194	0000 0000 0000 0000	194	0000 0000 0000 0000	194	0000 0000 0000 0000
195	0000 0000 0000 0000	195	0000 0000 0000 0000	195	0000 0000 0000 0000
196	0000 0000 0000 0000	196	0000 0000 0000 0000	196	0000 0000 0000 0000
197	0000 0000 0000 0000	197	0000 0000 0000 0000	197	0000 0000 0000 0000
198	0000 0000 0000 0000	198	0000 0000 0000 0000	198	0000 0000 0000 0000
199	0000 0000 0000 0000	199	0000 0000 0000 0000	199	0000 0000 0000 0000
200	0000 0000 0000 0000	200	0000 0000 0000 0000	200	0000 0000 0000 0000
201	0000 0000 0000 0000	201	0000 0000 0000 0000	201	0000 0000 0000 0000
202	0000 0000 0000 0000	202	0000 0000 0000 0000	202	0000 0000 0000 0000
203	0000 0000 0000 0000	203	0000 0000 0000 0000	203	0000 0000 0000 0000
204	0000 0000 0000 0000	204	0000 0000 0000 0000	204	0000 0000 0000 0000
205	0000 0000 0000 0000	205	0000 0000 0000 0000	205	

BAD ORIGINAL

LOC	ORG	LINE	SOURCE STATEMENT
0001	0000	220	JZ TIMING ;IF ZERO DOING 1ST CHRG. JNF
0002	0000	221	RLD R.#0F64 ;SEE IF DOING 9TH CHRG (CHRG=9)
0003	0000	222	JNE TIMING ;NO. JNF
0004	0000	223	JNF TIMING ;ARE JNF
		224	;
		225	THIS IS 4CHRG DISPLAY. SEE IF 1ST CHRG OF OTHER
		226	;
0005	0000	227	TIMING: RLD R.#0F64 ;TO TEST FOR CHRG COUNT
0006	0000	228	JNE TIMING ;LSN R2 IN R. JNF IF NOT ZERO
0007	0000	229	DEC R2 ;YES- DEC R2 (COUNT FOR #
		230	OF TIMES TO GO THEN 4CHRG DISPLAY
		231	WHEN 0 LAST TIME BEFORE DOING 9 CHRG
		232	REGD.
0008	0000	233	TIMING: RLD R.#0F64 ;GET R4 (CURRENT MAP REPRESENTATION
		234	OF REGD. DECIDED EARLIER). THEN R5
0009	0000	235	MOV R4 R
0010	0000	236	JNE R5 ;(LAST TIME CHARGE REGISTER FOR REGD.)
0011	0000	237	MOV R4 R
0012	0000	238	MOV R5 R
0013	0000	239	MOV R6 R
0014	0000	240	MOV R7 R
0015	0000	241	JNE TIMING ;IF R6=1 DOING 4 CHRG
		242	;
		243	TIMING DOES DISPLAY OF 9 CHRG ARE PRELIMINARY DETERMINING OF
		244	PERSONAL SWITCHES A-D
		245	;
0016	0000	246	TIMING: RLD R.#0F64 ;GET R2 AGAIN
0017	0000	247	ORL R1 #0 ;MAKE R1 = -1 TO TURN OFF DISPLAY
0018	0000	248	MOV R7 R ;A HAS CHRG. LSN IS REGD FOR 74145
0019	0000	249	MOV R8 R ;R1 HAS PTP TO CHRG (THAT ARE REGD
		250	ARE COPIES OF R2)
0020	0000	251	OUTL R1 R ;SEND CODED CHRG TO R1 FOR DISPLAY
0021	0000	252	CLF R ;GET REGD TO PERM. PIN (1) (=0)
0022	0000	253	MOV R5 R ;SEND TO 8243
0023	0000	254	CALL FIBOY ;DOES PRELIMINARY SW DEBOUNCING
0024	0000	255	MOV R6 R ;GET CHRG COUNT TO SEE IF DONE WITH
		256	PHASE 67 OR 85
0025	0000	257	RLD R.#0F64 ;276 + 005H = 0. IF DONE WITH CHRG '7',
0026	0000	258	JZ DONE11 ;JUMP
0027	0000	259	RLD R.#0F64 ;296 + 005H + 0F6H = 0 IF DONE
0028	0000	260	JZ DONE2 ;WITH CHRG '9', JNF
0029	0000	261	JNF TIMING ;ELSE JNF
		262	;
		263	CALL DONE WHICH DOES FINE
0030	0000	264	MOV R7 R ;DEBOUNCE STUFF
0031	0000	265	JF1 TIMING ;IF CHARGE F1=1. SAVE INFO IN R7(7)
0032	0000	266	RLD R.#0F64 ;NO CHANGE IN REGD?
0033	0000	267	TIMING: RLD R.#0F64 ;THIS IS LOOKED AT IN DONE3
0034	0000	268	MOV R6 R ;GET REGD FOR NEXT PHASE OF DEBOUNCING
0035	0000	269	JNF TIMING ;JNF TIMING
		270	;
		271	TIMING IS LIKE TIMING EXCEPT FOR 4 CHRG DISPLAY
		272	;
0036	0000	273	TIMING: RLD R.#0F64 ;MAKE R1=0 TO BLANK DISPLAY
0037	0000	274	MOV R5 R ;R6 HAS R2. LSN NIBBLE HAS ADDR FOR



LOC	OBJ	LINE	SOURCE STATEMENT
00FF 70D8		329	JF1 DONE30 ; ELSE IF CHANGE IN REGS9 JUMP
00FF BFD8		331	DONE37: MOV R7, #000H ; CLEAR OUT 1ST 6 BITS OF R7
		332	; WHERE THE EXTINTS, HOME AND CLUTCH
		333	; ARE. ANY TEST
		334	; WILL BE STOPPED. R7(6) = A FLAG
		335	; TO TELL WHEN DONE ALL AND R7(7)
		336	; WILL BE USED LATER DURING DONE1
		337	; AND DONE3, R2 IS SET UP TO 29.
00FE 1A		338	DONE33: INC R2
00FE FA		339	MOV R2, R2
00E8 30		340	MOV PS, A ; MOVE TO 74150 TO SET UP T1
00E8 FF		341	MOV R7, R7 ; GET R7
00E2 40D5		342	JNT1 DONE33 ; DON'T SET BIT UNLESS T1=1
00E4 17		343	INC A
00E5 77		344	DONE33: RR A ; ROTATE FOR NEXT BIT
00E5 1208		345	JBA DONE34 ; IF FIND PIT ALREADY SET, DONE-
00E5 AF		346	MOV R7, A ; ELSE MOV R7 TO A AND RELOOP
00E9 049E		347	JMP DONE33
00E5 77		348	DONE34: RR A ; DO 2 MORE ROTATES TO ALIGN
00E5 77		349	RR A
00E5 AF		350	MOV R7, A ; PUT BACK IN R7
00E5 3212		351	JBA DONE35 ; IF BIT 1 (EXTINT1) = 1, NEED TO
		352	; MODIFY BAKSTA
00C8 FF		353	DONE35: MOV R7, R7 ; NEED TO SEE IF SEMI-DEBOUNCE
00C8 1E		354	XRL R6, R6 ; CHG TO CLUTCH OR HOME. OLD R7
00C2 513F		355	ANL R6, #020H ; SAVED IN R6
00C4 060H		356	JZ DONE3F ; JUMP IF NO CHANGE TO HOME OR CLUTCH
00C2 2302		357	MOV R6, #PH02L ; SET BIT IN BAKSTA
00C8 74DE		358	CALL BAKP08
00C8 B520		359	DONE3F: MOV R2, #030H ; MAKE CHFF POINTER READY TO DO 4CHFF
00C1 B62E		360	MOV R0, #REGSW ; GET R0 READY FOR DEBOUNCE OF REGS5
00CE B004		361	MOV R3, #004H ; AND PUT IT'S # OF TIMES IN R3
00C0 04EP		362	JMP TIMRET
00C2 2304		363	DONE36: MOV R6, #10TSK ; HAVE I/O TASK TO SET UP IN BAKSTA
00C4 74DE		364	CALL BAKP08
00C6 B4C0		365	JMP DONE35
		366	DONE39: JB2 DONE38 ; DO K/B ONLY IF .00 + .000
		367	
		368	; JUMP IF .00
		369	; SEE IF .000
		370	; FOR EASE OF TEST
		371	; NO K/B IF JUMP
00D8 2308		372	DONE38: MOV R6, #KEYTSK ; CHANGE BAKSTA TO DO KEYBD TASK
00C4 74DE		373	CALL BAKP08
00C0 B4FC		374	JMP DONE37
		375	; ENDS OF TIMINT ROUTINE
		376	; NOO1SP IS FOR USE BY TIMINT TO KEEP THE ONE SHOTS FOR THE FAULT FFS
		377	; GOING. IF THERE IS NO DISPLAY (CHFF
		378	; POINTER NEGATIVE)
		379	; ONESHOT IS USED BY THE TIMINT ROUTINE DURING NORMAL DISPLAY
		380	; ACTION TO KEEP THE FAULT ONE SHOTS FIRING. AT THE END OF DONES
		381	; TIMRET IS USED AT THE END OF DONE1 AND DONE3
		382	; TIMFIN IS USED AT THE END OF ALL OTHER PATHS THRU THIS ROUTINE
		383	;
		384	; COME FROM DONES WITH FR=1, FR=0



LOC	OBJ	LINE	SOURCE STATEMENT
		385	;SAYS DOING NO DISPLAY, ONLY
		386	;PRINTER TIMER AND FAULT FF ONE SHOTS
		387	;
000E 85		388	WDTSP: CLR R8 ;SAYS DOING NO DISP, ONLY PRCTR
		389	;AND ONE SHOTS
000F 2387		390	ONESHT: MOV R, #MPST1 ;GET READY TO FIRE 2 ONE SHOTS
0011 3E		391	MOV R, R
0012 17		392	INC R ;TO MAKE CHANGE TO FAULT FFS
0013 3E		393	MOV R, R ;ONE SHOTS
0014 00EB		394	JFB TIMRET ;JUMP IF AT END OF DONES
0016 F9		395	MOV R, R1 ;ELSE GET OLD PORT 2
0017 3A		396	OUTL P2, R
0018 04EB		397	JMP TIMRET
001A 1A		398	TIMFIN: INC R2 ;INC CHARACTER POINTER
001B B929		399	TIMRET: MOV R, #SAVACC ;INIT R1 FOR LEAVING AND ENTERING
001D F1		400	MOV R, R1 ;INTERRUPT ROUTINES (USED TO
		401	;INDIRECTLY STORE AND FETCH BACKGROUND
		402	;ACCUMULATOR
001E 93		403	RETR ;DOES SEL PR8
		404	*****
		405	;
		406	;
		407	WRTDSP--S/R WHICH READS FROM RAM TO RAM (ACTUALLY INTENDED
		408	FOR READING A RAM REGISTER AND CONVERTING CHARS TO
		409	7 SEG CODE FOR 4 CHAR DISPLAY AND WRITING TO RAM).
		410	SEVERAL PARAMETERS ARE PASSED IT. R8 HAS WRITE ADDRESS
		411	(IN RAM), R1 HAS RAM ADDR (BRAM) FOR READ.
		412	FR=8 SAYS TO CPL CHARS (FOR 9 DIGIT DISPLAY)
		413	*****
		414	;
001F 18		415	WRTDSP: INC R8 ;INC RAM PTR (RAM PTR INCR BY
		416	;REDBR1)
0020 243F		417	CALL REDBR1 ;READ FROM 1ST RAM
0022 530F		418	ANL R, #CLRMSH ;GET READY TO DO MOVEP3 ON
		419	;BASIC OF CHAR--(DO TABLE LOOKUP)
		420	;OF PAGE 3
0024 F3		421	MOVEP? R, R ;GET ENCODED 7 SEG CODE FOR 4 CHAR
0025 B6F8		422	JFB WTDSP1 ;JUMP IF FOR 4 DIGIT
0027 37		423	CPL R
0028 0A		424	WTDSP1: MOV R, R ;PUT IN RAM
0029 EEEF		425	DJNZ R6, WTDSP
002B 83		426	RET ;RETURN WHEN DONE
		427	*****
		428	;
		429	DISFRG--THIS S/R DISENABLES THE FOREGROUND WHILE USING PORT 2
		430	FOR OTHER THINGS (MAINLY THE RAM)
		431	;
		432	*****
		433	;
002C 65		434	DISFRG: STOP TOUT
002D B91A		435	MOV R, #R2FORG ;FOR INDIR ACCESS OF R2
002E 74C4		436	CALL DSEMP6 ;MAKES P6 = F HEX, LEAVES IN A
0101 3F		437	MOV R, R
0102 F8		438	MOV R, R8 ;GET READY TO DISENABLE FOREGROUND
0103 4388		439	ORL R, #000H ;BY MAKING MSB OF R2 = 1



LOC OBJ	LINE	SOURCE STATEMENT
0105 R0	440	DSEPRG1: MOV R0, A
0106 55	441	STRT T
0107 B3	442	RET
	443	*****
	444	;
	445	WRITE-
	446	THIS S/P WRITES DATA TO THE RAM USING R0 AS AN ADDRESS
	447	& THE DATA IN THE LSN OF THE ACCUMULATOR. BITS 0-7
	448	OF R0 SELECT ONE OF 256 WORDS IN RAM-BITS P24 & P25
	449	OF PORT 2 DECODE AS FOLLOWS
	450	TO SELECT RAM 1 OR 2
	451	00- RAM 1 & RAM 2
	452	01- RAM 2
	453	10- RAM 1
	454	11- NEITHER RAM
	455	SEPARATE ENTRY PTS. ALLOWS WRITING TO RAM1 OR RAM2
	456	FIRST R0 IS WRITTEN TO P1 AND THEN CHECKED.
	457	AFTER WRITING THE WRITTEN DATA IS CHECKED
	458	THE RETURN IS MADE WITH R0 INC'D & THE ACCUMULATOR RESTORED
	459	R7 IS Clobbered BY THIS S/P. R0 IS RESTORED BY RETR. THE
	460	BACKGROUND MUST BE DISABLED. WRITE-110 M/S, WRITE-102.5 M/S.
	461	;
	462	*****
	463	;
0108 B5	464	WRITE1: CLR R0
0109 2412	465	JMP WRITE1
0108 B5	466	WRITE2: CLR R0
0109 55	467	CPL R0 ; R0-1 SAYS WRITING TO RAM 2
0108 530F	468	ANL R, #CLRMSH ; NEED TO WRITE TO BOTH RAMS
0108 FF	469	MOV R7, R ; SO NEED TO HAVE BOTH NIBBLES =
0110 47	470	SWAP R ; FOR RAM WRITE CHECK
0111 4F	471	ORL R, R7
0112 9A40	472	WRITE1: ANL P2, #SOLEH ; (040H) KEEP SOLEN SIGNAL
0114 8A1F	473	ORL P2, #PROG2F ; ENABLE 8243-2 SO CAN WRITE WRITE DATA TO P4
0116 3C	474	MOV P4, R
0117 9A4F	475	ANL P2, #SOLMF ; THIS PREPARES FOR WRITE TO BOTH RAMS
0119 B61D	476	JF0 WR0000 ; JUMP IF TO WRITE TO BOTH
0118 8A2F	477	ORL P2, #EBAM1 ; THIS WRITE ONLY TO RAM1
011D FF	478	WR0000: MOV R7, R ; SAVE ACCUMULATOR IN R7
011E F8	479	MOV R, R0 ; MOVE ADDRESS TO P1
011F 39	480	OUTL P1, R
0120 09	481	IN R, P1 ; CHECK THAT WROTE OUT TO P1 O.K.
0121 D8	482	XRL R, R0
0122 C626	483	JZ WR0001 ; JUMP IF O.K.
	484	FAULT:
0124 64EC	485	JMP FAULT1 ; SOFT FAULT, JUST TURN OFF MOTORS AND SOL AND STOP
0126 65	486	WR0001: STOP TOUT ; TO AVOID FOREGROUND CHANGING P2
0127 00	487	NOP ; TO ALLOW TIMER OVERFLOW TO HAPPEN IF READY
0128 B80F	488	ORL P2, #WRITEB ; (080H) DO WRITE
012A 9A7F	489	ANL P2, #SWRIBA ; (07FH) STOP WRITE & KEEP RAM MSB OF ADDRESS
012C 55	490	STRT T ; RESTART TIMER
012D 00	491	INS R, BUS ; READ RAM SELECTED
012E DF	492	XRL R, R7 ; COMPARE WRITTEN DATA WITH ACTUAL
012F B633	493	JF0 WRITE2 ; JUMP IF WROTE TO BOTH
0131 530F	494	ANL R, #CLRMSH ; ONLY INTERESTED IN LSN



LOC	OBJ	LINE	SOURCE STATEMENT
0132 C679	495	WRITEB: JZ	WRITEB ; JUMP IF READ WHAT WRITE
0135 238F	496	MOV	R, #00FH ; ELSE GET READY TO JUMP TO FAULT
0137 64E1	497	JMP	FAULT0
0139 FF	498	WRITEB: MOV	R, R7 ; RESTORE ACCUM
013A 18	499	INC	R0 ; INCREMENT THE CHAR POINTER
013B 93	500	PETR	
	501	*****	
	502	;	
	503	READB—THIS S/R READS THE RAM USING THE ADDR IN R1 & RETURNS THE	
	504	RESULT IN THE ACCUMULATOR	
	505	THERE ARE TWO ENTRY POINTS. READB1 WHICH READS FROM RAM 1	
	506	AND READB2 WHICH READS FROM BOTH RAMS. FB IS RESTORED BY RETR	
	507	P1 IS INCREMENTED AFTER THE READ. READB2 AND READB1—52.5 M/S.	
	508	READB1 — 57.5 M/S	
	509	;	
	510	*****	
	511	;	
013C 05	512	READB2: CLR	FB ; FB=0 SAYS READING FROM BOTH RAMS
013D 2441	513	JMP	READB1
013F 05	514	READB1: CLR	FB ; FB=1 SAYS READING FROM RAM 1
0140 95	515	CPL	FB
0141 9A4A	516	READB1: ANL	P2, #SOLEH ; (0A4A) KEEP SOLEH SIGNAL
0143 8A2F	517	ORL	P2, #EBR12
0145 F9	518	READB1: MOV	R, P1 ; PUT ADDR ON P1
0146 28	519	OUTL	P1, R
0147 09	520	IN	R, P1 ; CHECK THAT P1 = R1
0148 D9	521	XRL	R, R1
0149 9C24	522	JNZ	FAULT ; JUMP IF PROBLEM
014B 08	523	INS	R, BUS ; READ NIBBLE
014C 19	524	INC	R1 ; INCR THE CHAR POINTER
014D B55B	525	JFR	PORTS1 ; IF READING FROM BOTH RAMS DON'T JUMP
014F 93	526	RETR	
	527	; THIS SPECIAL COPY OF READB READS FROM THE WRITE ADDR	
	528	;	
0150 9A40	529	READB2: ANL	P2, #SOLEH ; KEEP SOLENOID
0152 8A2F	530	ORL	P2, #EBR11 ; READS FROM RAM1 ONLY
0154 F8	531	MOV	R, R0 ; GET ADDR FROM R0
0155 39	532	OUTL	P1, R ; PUT OUT TO P1
0156 09	533	IN	R, P1 ; CHECK THAT P1 = R0
0157 D8	534	XPL	R, R0
0158 9624	535	JNZ	FAULT
015A 08	536	INS	R, BUS ; READ DATA FROM BUS IN LSH
015B 530E	537	ORL	R, #10000000 ; CLEAR OUT MSH (FOR READB1)
015D 93	538	RETR	
	539	*****	
	540	;	
	541	S/R CLKVR — CLEAR KEYBOARD REGISTER IT DOES THAT AND	
	542	FIXES UP KEYSTA (IN R3) AND CALL CHKTOT. IT USES R, R0,	
	543	P1, R6, R7	
	544	;	
	545	*****	
	546	;	
015E B860	547	CLKVR: MOV	R0, #BKREG ; GET R0 READY FOR WRITEB1
0160 BE0A	548	MOV	R6, #00FAH ; GET R6 READY FOR LOOP CTR
0162 77	549	CLR	R ; WANT TO WRITE R TO REG

LOC	OBJ	LINE	SOURCE STATEMENT
0163	348C	550	CLR000: CALL WRTBM1 ;WRITE TO BAM
0165	EE63	551	DJNZ R6,CLR000 ;DONE YET?
0167	P92C	552	MOV R1,#REG29 ;FIX UP R3 (STATUS
		553	;OF KEYBOARD TASK) IN MSB REG29
0169	F1	554	MOV R1,R1
016A	FE	555	MOV R3,R1 ;OCE16 FIXES UP LSN
016E	DEP5	556	MOV R6,#025H ;TEMP FOR FOLLOWING
016D	F276	557	JBT CLR001 ;IF BIT7=1, P.O. + NORM + /CENTS
016F	CE	558	DEC R6 ;HENCE, ADD CHARS TO KBREG(1) VS (0)
0170	FE	559	CLR001: MOV R1,R6
0171	348C	560	CALL WRTBM1 ;WRITE TO KBREG(R1)
0173	841C	561	JMP CH7TOT ;MOVES DR TO TDR, RR TO TDR, ADDS
		562	;KE TO TDR, SURE KE FROM TDR.
		563	
		564	;ADD DR AND RR AN COMPARES WITH
		565	;TOTAL (COPY IN TEMP)
		566	;
		567	MOVREP--S/R WHICH MOVES A REGISTER FROM
		568	BAM TO ANOTHER REGISTER IN BAM
		569	
		570	R0 POINTS TO THE WRITE ADDR IN BAM
		571	R1 POINTS TO THE READ ADDR IN BAM
		572	THO ENTRIES POINTS ARE PROVIDED, MOVREP MOVES TO BAM1
		573	AND MOVREP MOVES TO BAM1 AND 2
		574	R6 AND R7 ARE BOTH Clobbered AS WELL AS THE ACCUM
		575	MOVREP TAKES 1.85 M/S (2.6 M/S INCLUDING FOREGROUND TIME)
		576	;
		577	;
0175	B5	578	MOVREP: CLR F0 ;F0 = 1 SAYS WRITE TO BAM1
0176	248C	579	JMP MREB1
0178	B5	580	MOVREP: CLR F0 ;THIS ENTRY POINT MOVES 3 REGS (29H) BYTES
0179	95	581	CPL F0
017A	BE29	582	MOV R6,#029H
017C	2492	583	JMP MREB1
017E	B5	584	MOVREP: CLR F0
017F	95	585	CPL F0 ;F0=1 SAYS WRITE TO BAM2
0180	FE	586	MOV R1,R0 ;1ST MAKE WRITTEN REGISTER DIRTY
0181	FE	587	MOV R6,R1 ;SAVE TO RESTORE R0 LATER
0182	430C	588	ORL R1,#009H ;THE 10TH NIBBLE MSB IS DIRTY BIT
0184	FE	589	MOV R0,R1
0185	67	590	DEC R1 ;TO GET LSN OF R1=0 (DIRTY BIT)
0186	B60C	591	JFB MREB2 ;JMP IF TO WRITE TO BAM2
0188	348C	592	CALL WRTBM1
018A	248C	593	JMP MREB3
018C	340C	594	MREB2: CALL WRTBM2
018E	FE	595	MREB3: MOV R1,R6
018F	FE	596	MOV R0,R1 ;GET BACK ORIGINAL R0
0190	DEP9	597	MOV R6,#009H ;GET LOOP CTR READY
0192	343F	598	MREB4: CALL REDEM1 ;READ FROM BAM USING R1
0194	BE9A	599	JFB MREB4 ;JMP IF TO WRITE TO BAM1 & 2
0196	340C	600	CALL WRTBM1
0198	249C	601	JMP MREB5
019A	340C	602	MREB4: CALL WRTBM2
		603	
019C	EE92	604	MREB5: DJNZ R6,MREB4 ;R0 AND R1 INCR BY WRTB AND REDEM1
			;DONE YET?



LOC	OBJ	LINE	SOURCE STATEMENT
019E 27		685	CLR A ; MUST CLEAR OUT DIRTY REGISTER BIT
019F 3488		686	CALL WRTBAB ; POTENTIAL KLUDGE HERE (BOTH)
01A1 F9		687	MOV A,R1 ; YES, REESTABLISH R1 FOR POSSIBLE
01A2 53FA		688	ANL A,#CLRASH ; REUSE BY CALLING ROUTINE
01A4 A9		689	MOV R1,A
01A5 F8		610	MOV A,R0 ; REESTABLISH R0 FOR POSSIBLE REUSE
01A6 53FA		611	ANL A,#CLRASH ; BY CALLING ROUTINE
01A8 A8		612	MOV R0,A
01A9 93		613	RETR
		614	*****
		615	;
		616	TOPTI—A S/R WHICH TURNS OFF THE PRINTER TIMER INTERRUPT BIT
		617	AND STARTS THE PRINTER TIMER
		618	;
		619	*****
		620	;
01AA 2384		621	TOPTI: MOV A,#STINTV
01AC B836		622	TOPTIC: MOV R0,#PRCTR
01AE 65		623	STOP TONT
01AF A8		624	MOV @R0,A ; RESTART PRINTER CTR
01B0 FA		625	MOV A,R2
01B1 529F		626	ANL A,#NPETTI ; TURN OFF TIMER BIT
01B3 AA		627	MOV R2,A
01B4 55		628	STRT T
01B5 83		629	RET
		630	*****
		631	;
		632	KBTLG—ROUTINE WHICH CHECKS IF THE KEYBOARD VALUE WILL CAUSE DR TO:
		633	1—BE NEGATIVE IF IN MODE WHICH ALLOWS TO USE ALL MONEY
		634	IN DR
		635	2—IF IN UPS MODE, NO CHECK DONE
		636	;
		637	*****
		638	;
01B6 B878		639	KBTLG: MOV R0,#TDRP8 ; THIS POINTS TO TDR(8) FOR LATER
01B8 2917		640	MOV R1,#METYPE ; GET COPY OF METYPE
01BA F1		641	MOV A,R1
01BB B2C1		642	JBS KBTLG4 ; JUMP IF UPS MODE
01BD 3458		643	CALL REDBMS
01BF 96C3		644	JNZ KBTLG3 ; JUMP IF BORROWED FROM
01C1 4458		645	KBTLG4: JMP KEYDF ; KB O.K.
01C3 B838		646	KBTLG3: MOV R0,#838H ; POINT TO 1ST CHR OF 4 CHR DISPLAY
01C5 1A		647	INC @R0 ; ADD OUT OF MONEY LIGHT
01C6 345E		648	CALL CLRKYR ; CLEAR THE KEYBOARD REG
01C8 34FB		649	CALL TOFFSS ; TURN OFF SOLENOID
01CA B975		650	MOV R1,#PRSTR ; CLEAR THE PRINTER STATUS
01CC A1		651	MOV @R1,A ; TOFFSS RETURN WITH A = 0
01CD 238E		652	MOV A,#88EH ; FOR KEYBOARD-FULL
01CF 4455		653	THP FINISH
		654	*****
		655	;
		656	SND—THIS S/R SENDS THE CURRENT COMMAND TO THE APPROPRIATE
		657	STEPPER
		658	;
		659	*****



LOC	OBJ	LINE	SOURCE STATEMENT
		660	;
01D1	34FA	661 SHD:	CALL TUPTI ; THIS CLEARS TIMER TASK BIT
		662	; AND RESTARTS PRINTER TIMER
01D3	FC	663	MOV R,R4 ; GET CURR COMMAND TO R7 (TEMP)
01D4	47	664	SWAP R
01D5	AF	665	MOV R7,R
01D6	9A48	666	ANL P2,#SOLEH ; SAVE SOLENOID
01D8	8A28	667	ORL P2,#PROG1 ; GET READY TO SEND OUT STEPPER COMMANDS
01DA	FB	668	MOV R,P3 ; SEE WHICH STEPPER TO SHD TO
01DE	32F5	669	JR1 SHD2
01DD	12E2	670	JR0 SHD1
01DF	FF	671 SHD0:	MOV R,R7 ; GET STEPPER COMMAND
01EA	3C	672	MOVD P4,R ; SEND TO IT'S 8279 PORT
01E1	83	673	RET
01E2	FF	674 SHD1:	MOV R,R7
01E3	3D	675	MOVD P5,R
01E4	83	676	RET
01E5	12EA	677 SHD2:	JR0 SHD3
01E7	FF	678	MOV R,R7
01E8	3E	679	MOVD P6,R
01E9	83	680	RET
01EA	FF	681 SHD3:	MOV R,R7
01EB	3F	682	MOVD P7,R
01EC	83	683	RET
		684	*****
		685	;
		686	;
		687	NORINT—THIS IS THE NORMAL INTERRUPT S/R. IT TURNS OFF
		688	THE DISPLAY AND SOLENOID AND STEPPERS AND FIXES
		689	UP THE STATE TO POWER DOWN GRACEFULLY;
		690	;
		691	*****
		692	;
		693	; COME HERE BECAUSE PARLSS = 1 (/INT = 0)
		694	; ALREADY DONE SEL RB1 AND MOV R6,R (SAVED THE ACCUM)
		695	;
01ED	15	696 NORINT:	DIS I ; TURN OFF INTERRUPT
01EE	14FC	697	CALL DISFRG ; DISABLE THE DISPLAYS
		698	; (JUST KEEPS UP THE DECR OF PRCTR
		699	; AND THE FIRING OF
		700	; THE ONSHOTS FOR THE FAULT FFS)
01F0	34F8	701	CALL TOFFSS ; TURN OFF STEPPERS AND SOLENOID
01F2	B802	702	MOV R0,#BAKSTA ; WANT TO MAKE BAKSTA(7)=1
01F4	B808	703	MOV R0B,#PFTSK
01F6	FE	704	MOV R,R6 ; GET BACK ACCUM AND RETURN
01F7	93	705	RETR
01F8	8A48	706 TOFFSS:	ORL P2,#SOLEH ; TURN OFF SOLENOID
01FA	9A48	707 TOFFST:	ANL P2,#SOLEH ; KEEP SOLENOID IF USED BY PRINT
01FC	8A28	708	ORL P2,#PROG1 ; GET READY TO TURN OFF STEPPERS
01FE	27	709	CLR R
01FF	3C	710	MOVD P4,R ; 0 TO THE STEPPERS TURNS OFF THEIR POWER
0200	3D	711	MOVD P5,R
0201	3E	712	MOVD P6,R
0202	3F	713	MOVD P7,R
0203	83	714	RET

LOC	OBJ	LINE	SOURCE STATEMENT
		715 ;	*****
		716 ;	
		717 ;	
		718 ;	KEYBD—THIS CO-ROUTINE IS JUMPED TO FROM MAIN IF BAKSTA(3)
		719 ;	=1 THE FLOW THRU THIS ROUTINE IS CONTROLLED BY
		720 ;	THE LSH OF BAKSTA IT IS CALLED THE CURRENT
		721 ;	CHARACTER AND IS = TO THE CURRENT KEYBOARD SWITCH
		722 ;	BEING PRESSED. SOMETIMES IT CONTAINS INFO ABOUT
		723 ;	SWITCHES THAT HAVE BEEN PRESSED. IF NOTHING
		724 ;	DEPRESSED, CC=10 (0AH), CC=12 SAYS DISPLAYING ALL SEGS,
		725 ;	CC=14 SAYS KEYBOARD REG FULL
		726 ;	
		727 ;	*****
		728 ;	
		729 ;	;R3 IS INITED WITH KEYSTA
		730 ;	
0204 14FC-		731 KEYBD:	CALL DISFRG ;DISABLE FOREGROUND DURING KEYBOARD
0206 B92A		732 MOV	RL, #REG87 ;INIT RL TO REG 807
0208 F1		733 MOV	R, R01
0209 FE		734 MOV	R6, R ;PLACE IN R6
020A B92C		735 MOV	RL, #REG89 ;INI FOR LATER
020C 85		736 CLR	F0 ;FOR LATER USE
020D FB		737 MOV	R, R3 ;GET KEYSTA
020E 7212		738 JBS	KEY000
0210 441B		739 JMP	DISCHA ;IF 7 OR LESS NO SPEC STATUS
0212 5216		740 KEY000:	JBS KEY001
0214 441B		741 JMP	DISCHA ;ITS 8 TO 11, NOT SPEC STATUS
0216 321A		742 KEY001:	JBS KEY002
0218 441F		743 JMP	DISPAL ;CC=12, SEE IF STILL 7+8+9
021A 95		744 KEY002:	CPL F0 ;CC=14 SAVE IN F0
		745 ;	
		746 ;	;DISCHA IS DISCOVER CHARACTER IT LOOKS FOR A CHAR IN THIS ORDER
		747 ;	;7, 8, 9, 4, 5, 6, 1, 2, 3, CLR BATCH, 0, CLR KEYBD. THIS IS DONE TO TRY TO
		748 ;	;AVOID THE FAT FINGER PROBLEM WITH A RIGHT-HANDED PERSON RL HAS
		749 ;	;ADDR OF REG89
		750 ;	
021B F1		751 DISCHA:	MOV R, R01 ;GET REG89
021C 2E		752 XCH	R, R6 ;R6 HAS REG87 IN IT
021D BF07		753 MOV	R7, #007H ;R7 CARRIES BCD CHAR BEING CHECKED
021F E276		754 JBS	CHK289 ;IF CHAR 7 DO WE HAVE 8 AND 9 ALSO?
0221 2E		755 XCH	R, R6 ;ELSE LOOK TO SEE IF EIGHT OR NINE
0222 1F		756 INC	R7 ;7 CONTAINS THE BCD OF THE CHAR BEING
		757	;CHECKED
0223 1267		758 JBS	FINNO ;JMP IF '8'
0225 1F		759 INC	R7
0226 3267		760 JBS	FINNO ;JMP IF '9'
0228 2E		761 XCH	R, R6 ;LOOK AT 4, 5, 6, ETC
0229 BF04		762 MOV	R7, #04
022B 9267		763 JBS	FINNO
022D 1F		764 INC	R7
022E B267		765 JBS	FINNO
0230 1F		766 INC	R7
0231 D267		767 JBS	FINNO
0233 BF01		768 MOV	R7, #001H
0235 3267		769 JBS	FINNO



LOC	OBJ	LINE	SOURCE STATEMENT
0237	1F	770	INC R7
0238	5267	771	JB2 FINND
0239	1F	772	INC R7
023E	7267	773	JR3 FINND
023D	2E	774	XCH R, R6 ; GET REG8 TO CHECK CLEAR BATCH
023E	7292	775	JB3 FINSAT ; CLEAR BATCH IS DEPRESSED
0240	2E	776	XCH R, R6 ; CHECK BIT 'A'
0241	BFB6	777	MOV R7, #000H
0243	1267	778	JB6 FINND
0245	2E	779	XCH R, R6 ; CHECK CLEAR KEYBOARD
0246	52E4	780	JB2 FNCLYS
0248	B82D	781	MOV R0, #1500H ; NO KEYBOARD SW. DOWN. SEE IF CHANGE TO
024A	FB	782	MOV R, R6 ; P.O. MODE
024B	92E4	783	JB4 FNCLYS ; JUMP IF YES
024D	FB	784	MOV R, R3 ; NOTHING IS DEPRESSED
		785	; IF CC=10 OR LESS, MAKE CC=10.
024E	0386	786	ADD R, #000H ; OTHER WISE LEAVE THE SAME
0250	C7	787	MOV R, PSW
0251	D258	788	JB6 KEYDF ; JUMP IF AUX CARRY = 1
0253	238A	789 CCE10:	MOV R, #000H ; MAKE CC=10
0255	B803	790 FINND:	MOV R0, #003H ; INIT R0 FOR XCHD
0257	3F	791	XCHD R, R0
0258	FB	792 KEYDF:	MOV R, R3
0259	B834	793	MOV R0, #KEYSTR ; REPLACE KEYSTR WITH R3
025B	FA	794	MOV R0, R
025C	B835	795	MOV R0, #PRSTR ; IF DOING CONSECUTIVE ENVELOPES.
025E	FB	796	MOV R, R0 ; PRSTR WILL NO BE ZERO
025F	D667	797	JZ MAINJ2 ; JUMP IF NOT DOING CONSEC ENVS
0261	E479	798	JMP MAINR2 ; ELSE JUST GO BACK TO MAIN
0263	23F2	799 MAINJ2:	MOV R, #KEYTOP ; GET READY FOR MAIN.
0265	E476	800	JMP MAINR
		801	;
		802	; FOUND A NUMBER KEY DEPRESSED, IT'S BCD CODE IS IN R7, IF
		803	; CC=8-9, IT STAYS THE SAME VALUE, IF CC=10 THEN CC= NEWVALUE
		804	; CC=12 NOT POSSIBLE HERE, AND CC=14 AVOIDS THIS CODE.
0267	B6E3	805 FINND:	JF8 MAINJ2 ; IF CC=14 NOTHING TO DO
0269	FB	806	MOV R, R3
026A	538F	807	RAL R, #CLRASH
026C	03F6	808	ADD R, #0F6H ; 0F6H + 0F6H = 0
026E	9658	809	JNZ -KEYDF ; JUMP IF NOT = TO 0F6H
0270	FF	810	MOV R, R7 ; CC=10, SO MAKE CC=NEW CHAR
0271	B803	811 FINND1:	MOV R0, #003H ; R7 HAS IT, GET READY FOR XCHD
0273	30	812	XCHD R, R0
0274	6431	813	JMP RKBREG
		814	;
		815	; CHA789 CHECK IF HAVE 7*0*9, ALREADY HAVE FOUND '7', R7 HAS 07H IN
		816	; IT FOR FINND AND R6 HAS REG 09
		817	;
0276	2E	818 CHA789:	XCH R, R6 ;
0277	37	819	CPL R ; FOR ERASE OF TESTING
0278	1267	820	JB8 FINND
027A	3267	821	JB1 FINND
027C	27	822	CLR R
027D	BFB9	823	MOV R7, #000H ; DISPLAY ALL SEGMENTS, INIT ACCUM
027F	B920	824	MOV R1, #020H ; TO DISPLAY ALL CHARS IN 9 CHAR DISP
			; R7 IS LOOP CTR, R1 IS POINTER TO



LOC	OBJ	LINE	SOURCE STATEMENT
		825	;DECODED 7SEG CODES FOR 9 CHAR
0201	RI	826	DISALP: MOV RCL, A ;PUT OUT CHAR SEGMENTS
0202	19	827	INC R1
0203	FE01	828	DJNZ R7, DISALP ;DONE?
		829	;SEE IF A = 0 OR -1 IF -1
		830	;READY TO RETURN, ELSE, MUST DO
		831	;4CHAR DISPLAY
0205	F20E	832	JB7 DSALP1
0207	B930	833	MOV R1, #030H ;READY TO DISPLAY ALL SEGS FOR 4CHAR
0209	BF04	834	MOV R7, #004H ;STARTS AT 030H, HAS 4 CHARS,
020B	07	835	DEC A ;NEEDS -1 TO DISPLAY ALL
020C	4481	836	JMP DISALP
020E	230C	837	DSALP1: MOV R, #00CH ;MAKE A = 12 FOR DC=12
		838	;WILL CLEAR KB AT END OF 7*8*9
0200	4455	839	JMP FINH08
		840	;
		841	;FINH08—CAME HERE BECAUSE FOUND CLR BATCH DEPRESSED.
		842	;IF P.O. MODE WE ARE READY TO ADD OR SUB FROM DREG.
		843	;IF NOT P.O. LOOKING FOR MOMENTARY DEPRESSION OF CLEAR KB
		844	;TO INDICATE IT'S REALLY CLEAR BATCH IN EITHER CASE, FB SAYS CLR KB
		845	;WAS DEPRESSED WHILE CLR BATCH DOWN FB = 1 IN P.O. SAYS SUBTRACT,
		846	;FB IN /P.O. SAYS DO CLEAR BATCH NO FB SAYS DO NOTHING
		847	;
0232	05	848	FINH08: CLR FB ;USE FB AS FLAG TO TELL WHETHER
		849	;TO ADD OR SUB K/R FROM DREG
		850	;R1 IS INITED TO REG09 WHICH WE NEED
		851	;TO READ TO SEE CHANGES IN CLR BATCH
		852	;AND CLR KEYBOARD
0293	74D5	853	CALL ENHFRG ;NEED TO SEE CHANGES IN KEYBOARD
0295	F1	854	POBATL: MOV R, R01 ;GET NEW COPY OF SWITCHES
0296	B690	855	JF0 POBAT1 ;IF FB ALREADY SET DON'T CHECK
0298	529C	856	JB2 POB000 ;IS CLRKEYBD = 1
029A	4450	857	JMP POBAT1 ;NO
029C	95	858	POB000: CPL FB ;YES, MAKE FB = 1
029D	7295	859	POBAT1: JB3 POBATL ;IS CLR BAT RELEASED?
029F	14FC	860	CALL DISFRG ;YES, DISABLE FOREGROUND AGAIN
02A1	FB	861	MOV R, R3 ;SEE IF P.O. MODE
02A2	37	862	CPL A ;FOR ERASE OF JMP
02A3	9205	863	JB4 CLRBAT ;JMP IF NOT
02A5	B870	864	MOV R0, #BTDR06 ;THIS IS P.O. ADD/SUB
02A7	74A0	865	CALL ARKBNC ;ADD/SUB TDREG AND KBD TO TDREG
02A9	C8	866	DEC R0 ;R0 POINTS TO TDREG(9), WANT TDREG(8)
02AB	3450	867	CALL RET005 ;IF NOT ZERO, TDREG IS GTR THAN
02AC	C600	868	JZ POADD1 ;\$100,000 OR LESS THAN \$0.0
02AE	24C3	869	JMP KBTLG3 ;THIS LIGHTS OUT OF MONEY AND CLRS KB
02B0	B000	870	POADD1: MOV R0, #BT00P ;HAS COPY OF TOTAL IN IT
02B2	74A0	871	CALL ARKBNC ;ADD/SUB KEYBOARD TO THAT
02B4	B990	872	MOV R1, #BT00P
02B6	B000	873	MOV R0, #BT00P ;GOING TO GET BT00P(BANK2) = TOTAL
02B8	3470	874	CALL MOV005 ;+/- KEYBOARD
02BA	05	875	CLR FB ;NOW WANT TO MAKE BT00P(BANK2) =
02BB	B000	876	MOV R0, #BT00P ;BT00P + (BT00P +/- KEYBOARD)
02BD	B970	877	MOV R1, #BT00P
02BF	74A0	878	CALL ARKBNC
02C1	B000	879	MOV R1, #BT00P

LOC	OBJ	LINE	SOURCE STATEMENT
B2C3	B896	880	MOV R6, #BTEMP ; NOW MOVE RESULTS TO BTEMP1
B2C5	3475	881	CALL MOVRE1
B2C7	B896	882	MOV R1, #BTEMP ; CHECK THAT BTEMP 1 AND 2 =
B2C9	748A	883	CALL COMPB ; CHECK THAT THE SAME METER WITH
B2CE	D433	884	CALL MYTDR ; IF O.K., MOVE TDR TO DR
B2D1	B896	885	MOV R1, #BTEMP ; MOVE BTEMP TO BTOTAL
B2DF	B826	886	MOV R6, #BTOTAL ; CHECKS IF THESE MOVES IS DONE BY
B2D1	347E	887	CALL MOVRE2 ; CH.TOT IN CLKVR
B2D3	44E4	888	JMP FNCLK5 ; JUMP TO CLEAR KEYBOARD REG
		889	;
		890	; THIS IS CODE TO CLEAR BATCH REGISTERS (IF F0=1)
		891	;
B2D5	95	892	CLPRT: CPL FR ; FOR ERASE OF TEST
B2D6	B6E3	893	JF0 MAINJ2 ; JUMP IF F0 WAS 0
B2D8	345E	894	CRBAT1: CALL CLKVR ; CLEARS REGS AND MOVES REGS TO TEMPS
B2DA	B838	895	MOV R6, #CBAT1
B2DC	B966	896	MOV R1, #CBAT2REG ; TO CLEAR BATCH REGS, WILL MOVE
B2DE	3475	897	CALL MOVRE1 ; CLEARED KB REG TO BATCH TOTAL
B2E0	B84F	898	MOV R6, #BRECT ; AND COUNT
B2E2	3475	899	CALL MOVRE1 ; ONLY 1 COPY OF EACH
B2E4	345E	900	FNCLK5: CALL CLKVR ; MAY BE REDUNDANT
B2E6	740C	901	CALL REGFER ; MAY ALSO BE REDUNDANT
B2E8	FE	902	MOV R1, R3 ; SEE IF P.O. MODE = 1
B2E9	9253	903	JB4 CDE10 ; NO
B2EB	948E	904	CALL DISKBE ; DISPLAY THE KEYBOARD REGISTER THEN
B2ED	4453	905	JMP CDE10 ; END
		906	;
		907	; DISPR-- DISPLAYING R1 SEGS. SEE IF STILL 7 + 8 + 9
		908	; R6 HAS #REG97, R1 HAS PTR TO REG89
		909	;
B2EF	F1	910	DISPR: MOV R1, R3 ; GET REG89
B2F0	1258	911	JB0 KEYDF ; STILL '8' DOWN
B2F2	3258	912	JB1 KEYDF ; STILL '9' DOWN
B2F4	2E	913	XCH R1, R6 ; GET REG87
B2F5	F258	914	JB7 KEYDF ; STILL '7' DOWN
B2F7	44E4	915	JMP FNCLK5 ; TO DISPLAY REGISTERS
		916	;
		917	; THIS IS IN THE 4TH PAGE OF PROGRAM MEMORY, WHICH STARTS WITH THE
		918	; ENCODE FROM BCD TO 7SEGMENT DISPLAY
		919	;
B300		920	ORG 300H
B300	FC	921	CHAREN: DB 0FCH
B301	60	922	DB 050H
B302	DA	923	DB 00AH
B303	F2	924	DB 0F2H
B304	66	925	DB 066H
B305	B6	926	DB 066H
B306	BE	927	DB 08EH
B307	E0	928	DB 0E0H
B308	FE	929	DB 0FEH
B309	E6	930	DB 0E6H
		931	;
		932	;*****
		933	;
		934	; COMPBA-- COMPARES EITHER A REGISTER OR ONE NIBBLE BETWEEN THE



LOC	OBJ	LINE	SOURCE STATEMENT
		935 ;	TWO BYTES, OR AS MANY NIBBLES AS WANT IF USE CXPLOT
		936 ;	IF ENTER AT COMPEN: F1 IS SET, TELLS WHICH TYPE OF FAULT
		937 ;	IF GET NOT COMPARE F1=1 SAYS DOING MATH COMPARE = 0
		938 ;	SAYS DOING MOVE COMPARE
		939 ;	
		940 ;	*****
		941 ;	
030A	RS	942 COMPEN: CLR	F1
030B	RS	943 CPL	F1 ; SIGNAL TO FAULT STUFF IN THIS ROUTINE
030C	6416	944 JNP	CHFB00
030E	EE01	945 COMPEN: MOV	RG, #001H ; GET READY TO DO NIBBLE
0310	RS	946 CXPLOT: CLR	F1
0311	RS	947 CLP	FA
0312	95	948 CPL	F0 ; SAYS TO NOT CHANGE R1 AT END
0313	6419	949 JNP	CHFB01
0315	RS	950 COMPEN: CLR	F1
0316	EE0A	951 CHFB00: MOV	RG, #00AH ; GET READY TO DO REGISTER
0318	RS	952 CLR	F0 ; SAYS TO MAKE R1 POINT TO
		953	; BEGINNING OF REGISTER COMPARED
0319	343C	954 CHFB01: CALL	RED0AB
031E	RD	955 MOV	RS, A ; SAVE FIRST RESULT IN RS
031C	47	956 SWP	A ; RESULTS OF READ IN BOTH NIBBLES
		957	; SHOULD BE SAME, SWP AND XRL
031D	DD	958 XRL	A, RS ; COMPARE
031E	6627	959 JZ	CHFB03 ; ERROR IF NOT JUMP
0320	2366	960 MOV	A, #006H
0322	7625	961 JF1	CHFB04
0324	17	962 INC	A ; INCR A IF MOVE COMPARE ERROR
0325	64E1	963 CHFB04: JNP	FAULT0
0327	EE19	964 CHFB03: DJNZ	RG, CHFB01
0329	B62F	965 JFA	CHFB02
032B	F9	966 MOV	A, R1 ; F0=0, CHANGE R1 TO REG(0)
032C	53F8	967 ANL	A, #CLR15H
032E	0A	968 MOV	R1, A
032F	FD	969 CHFB02: MOV	A, RS ; GET LAST VALUE INTO A
0330	92	970 RETR	
		971 ;	*****
		972 ;	
		973 ;	AKBREG—ROUTINE WHICH ADDS NEW CURR CHAR TO KEYBOARD REGISTER
		974 ;	AND THEN COMPUTES IF THE REGISTER IS FULL AND THEN
		975 ;	DISPLAYS THE NEW REGISTER DEPENDING UPON P.O. MODE
		976 ;	IF NOT P.O. ALSO GETS NEW COPIES OF TDREG AND TAREG
		977 ;	FROM AREG AND DREG USING CHKTOT THEN ADDS KB REG TO TAREG
		978 ;	AND SUBS KB REG FROM TDREG AND CHECKS THAT TAREG + TDREG
		979 ;	= A TEMP COPY OF TOTAL BROUGHT OVER BY CHKTOT, THEN DISPLAYS
		980 ;	NEW KEYBOARD REG AND CHECKS THAT THE KEYBOARD REG VALUE ISN'T
		981 ;	TOO LARGE
		982 ;	THE MOST SIGNIFICANT BIT OF THE KEYBOARD REGISTER (AH)
		983 ;	CONTAINS THE NUMBER OF CHARACTERS IN THE KEYBOARD
		984 ;	REG + 5. THIS IS FOR EASE OF TESTING ALSO MUST BE
		985 ;	DECIDED WHERE TO ADD THE NEW CHARACTER (POSITION 1 OR 0).
		986 ;	THIS IS DECIDED ON THE BASIS OF R3(7), WHICH IS P.O.
		987 ;	MODE + FCENTS*HORN
		988 ;	
		989 ;	*****



LOC	OBJ	LINE	SOURCE STATEMENT
		990	;
		991	;USES R0, R1, R6, R7, R4, R5
		992	;
0331	R050	993	AKREG: MOV R0, #02CH5 ;INI R0 WITH ADDR OF CURR KB CHAR
0333	3450	994	CALL REDRMS ;GET IT
0335	17	995	INC A ;INCREMENT IT
0336	RE	996	MOV R4, A ;SAVE IT FOR LATER
0337	3400	997	CALL WRTBM1 ;WRITE IT TO BMM
0339	B009	998	MOV R6, #009H ;USE R6 AS LOOP COUNTER FOR LEFT
0330	FE	999	AKREG1: MOV R, R6
0330	4300	1000	ORL A, #BIT0REG ;SHIFT OF KEYBOARD REGISTER
0330	RE	1001	MOV R0, A ;INI R0 AND R1 FOR WRITE AND READ
0330	RE	1002	MOV R1, A ;WRITE ADDR 1 GTP THEN READ ADDR
0340	C9	1003	DEC R1
0341	343F	1004	CALL REDRMS ;GET CHAR
0342	3400	1005	CALL WRTBM1 ;WRITE IT BACK
0345	EE30	1006	DJNZ R6, AKREG1 ;DONE YET?
0347	C0	1007	DEC R0 ;TO OVERCOME AUTO INCR IN WRTBM1
0340	FE	1008	MOV R, R3 ;IF P.O. + NORM + /CENTS ADD TO
0349	F24C	1009	JR7 AKREG2 ;KBDREG(1) OTHERWISE KBDREG(0)
0340	C0	1010	DEC R0 ;THE NEW CHAR GOES INTO (0)
0340	3400	1011	AKREG2: CALL WRTBM1 ;WRITE OUT NEW CHAR
		1012	;
		1013	;SEE IF KEYBD REG IS FULL
		1014	;
034E	FC	1015	MOV R, R4 ;IF LESS THAN 3 (5+X CHARS) NOT FULL
034F	7270	1016	JR3 AKREG7 ;JUMP IF MORE THAN 3 CHARS
0351	740C	1017	AKREG4: CALL REGREQ ;POSSIBLY REDUNDANT
		1018	;ALSO COME HERE FROM END OF PRINT
0353	FB	1019	MOV R, R3 ;SEE IF P.O. MODE = 1
0354	5270	1020	JR4 AKREG5 ;JUMP IF YES
0356	9400	1021	CALL CHKTOT ;GET NEW COPIES OF DR AND AR
0358	B000	1022	MOV R0, #BTAREG ;FB IS CLEARED BY CHKTOT
0359	7400	1023	CALL ARKBN ;ADD KB TO TDR
0350	95	1024	CPL FB ;TO DO SUBTRACT
0350	B070	1025	MOV R0, #BTAREG
035F	7400	1026	CALL ARKBN ;SUBTRACT KB FROM TDR
0361	B090	1027	MOV R0, #BTEMP ;NOW MUST CHECK TAREG + TAREG
0363	B900	1028	MOV R1, #BTAREG ;= COPY OF TOTAL IN BTEMP(BAM2)
0365	3475	1029	CALL MOVRE1 ;MOVE TAREG TO TEMP(BAM1)
0367	B090	1030	MOV R0, #BTEMP
0369	B970	1031	MOV R1, #BTAREG
0360	05	1032	CLP FB ;WANT TO ADD TAREG AND TAREG (IN TEMP)
0360	7402	1033	CALL ARANC ;DO ADD
036E	B990	1034	MOV R1, #BTEMP ;DO COMPARISON
0370	7400	1035	CALL COMPM ;FOR MATH
0372	9400	1036	CALL DISK0P ;SO DISPLAY NEW KEYBD REG ON 4CHAR
0374	2400	1037	JMP KBTLG ;JUMP TO SEE IF KEYBOARD TOO LARGE
0376	4450	1038	AKREG5: JMP KEYDF
		1039	;
		1040	;SEE IF KEYBD REG FULL. IF P.O. MODE, NEED TO SEE IF UP TO
		1041	;MAX P.O. VALUE, ENCODED IN LSN OF MTYPE
		1042	; IF NOT P.O., IF FCENTS=1 OR IF 3 STEPPER METER
		1043	;THEN WE ARE NOW FULL OTHERWISE IF 99.99 (BIT0=1), IT'S FULL
		1044	;

LOC	OBJ	LINE	SOURCE STATEMENT
0378	B817	1845	AKREG7: MOV R0, #METYPE ; INIT R0 TO GET MAX P.O VALUE AND
037A	FB	1846	MOV A, R3
037B	928C	1847	JBA AKREG8 ; JUMP IF P.O. MODE
037D	D285	1848	JBE AKREG9 ; JUMP IF EVENTS
037F	FC	1849	MOV A, R4
0380	1285	1850	JBA AKREG9 ; OR THIS TO MAKE KEYBD REG FULL
0382	FA	1851	MOV A, R08 ; LOOK TO SEE IF 3 OR 4 STEPPERS
0383	F251	1852	JBT AKREG4 ; JUMP IF 4
0385	D883	1853	AKREG9: MOV R0, #003H ; GET READY TO MAKE LSH OF R3
0387	238E	1854	MOV A, #02EH ; = 140 USING XCHD
0389	38	1855	XCHD A, R08
038A	6451	1856	JMP AKREG4
038C	FA	1857	AKREGA: MOV A, R08 ; GET MAX P.O. VALUE
038D	DC	1858	XRL A, R4 ; DO XOR WITH CURR KBCC
038E	538F	1859	ANL A, #CLRMSH
0390	C685	1860	JZ AKREG9 ; IF FULL, JUMP
0392	6451	1861	JMP AKREG4 ; ELSE JUMP HERE TO END
		1862	; *****
		1863	;
		1864	DONE — S/R WHICH DOES THE END COMPARISON OF THE FOREGROUND
		1865	REGISTERS R4, R5, R6 TO DO THE DEBOUNCING OF SWITCHES.
		1866	IT IS CALLED AFTER THE REGISTERS HAVE BEEN SET
		1867	UP BY FDBCK AND THEY ARE REALIGNED. THE LOGIC FOR ONE BIT
		1868	IS AS FOLLOWS: AN XOR IS DONE BETWEEN THE CURRENT
		1869	VALUE (TO THE M/P) IN R4 AND THE NEW VALUE (GOTTEN
		1870	FROM T1). THIS IS DONE BY FDBCK. IN R5 IS THE VALUE
		1871	OF THIS COMPARISON LAST TIME THE SWITCH WAS CHECKED.
		1872	THE REST OF THIS IS DONE BY THIS S/R. NOW THE LAST
		1873	COMPARISON IN R5 IS COMPARED WITH THIS TIME COMPARISON
		1874	IN R6. SO IF THERE WAS A CHANGE NOTICED BOTH TIMES,
		1875	THIS LAST COMPARISON (AN AND) WILL BE TRUE. THIS
		1876	VALUE IS NOW SAVED IN R5 (FOR LATER USE BY TIMINT)
		1877	IF IT IS NONZERO. ALSO F1 IS SET TO INDICATE THIS.
		1878	NOW R4 (THE CURRENT M/P VIEW OF THE SWITCH) IS XORED
		1879	IF THE VALUE IN R5 IS NON ZERO. ANY OF THE SWITCHES
		1880	WITH THEIR EQUIVALENT BIT IN R5 BEING TRUE WILL BE
		1881	CHANGED. FROM 1 TO 0 OR V.V. R6 THE NEW LAST TIME
		1882	CHANGE VALUES ARE STORED USING R08 AND IF F1 WAS
		1883	SET, THE NEW CURRENT VALUES OF THE DEBOUNCED SWITCHES
		1884	ARE STORED BACK. AFTER DECREMENTING R0. IN EITHER
		1885	CASE, THE NEW CURRENT VALUES OF THE DEBOUNCED SWITCHES
		1886	ARE RETURNED IN THE ACCUM WITH R0 POINTING TO WHERE
		1887	THEY SHOULD BE STORED IF NECESSARY TO MAKE A CHANGE
		1888	TO THESE REGISTERS BY THE DONEX ROUTINES.
		1889	;
		1890	; *****
		1891	;
0394	FE	1892	DONE: MOV A, R6 ; GET NEW LAST TIME CHANGES
0395	80	1893	MOV @R0, A ; STORE IT
0396	C8	1894	DEC R0 ; NOW POINTS TO M/P CURRENT VALUE
0397	5D	1895	ANL A, R5 ; DO AND BETWEEN THIS TIME CHANGES
		1896	; AND LAST TIME CHANGES
0398	R5	1897	CLR F1
0399	C69C	1898	JZ FNCHG ; JUMP IF AND = 0, NO CHANGES
039C	R5	1899	CP1 F1 ; ELSE COMPLEMENT F1 TO 1



LOC	OBJ	LINE	SOURCE STATEMENT
029C	R0	1100	FNCHG: MOV R5,R ;SAVE THE VALUE IN THE ACCUM IN R5
		1101	;FOR LATER USE BY TIMINT
029D	DC	1102	XPL R,R4 ;DO XOP
029E	RA	1103	MOV R0,R ;STORE NEW CURRENT VALUES
039F	83	1104	RET
		1105	*****
		1106	;
		1107	APRNC—S/R WHICH ADDS A REGISTER IN R0 TO THE KEYBOARD REGISTER
		1108	IF F0 = 1, IT SUBTRACTS THE KEYBOARD REGISTER FROM THE R0
		1109	REGISTER APRNC TAKES 2.2 M/S (3.0 M/S INCL. FOREG)
		1110	;
		1111	*****
		1112	;
03A0	E950	1113	APRNC: MOV RL,#KBREG ;INIT R1 TO THE KEYBOARD REG
03A2	95	1114	APRNC: CPL F0 ;LET F0 = 1 MEAN ADD
03A3	97	1115	CLR C ;CLEAR CARRY BIT
03A4	B6A7	1116	JF0 R2 ;SET IF SUBTRACTING
03A6	A7	1117	CPL C
03A7	BE00	1118	AP2: MOV R6,#009H ;INIT LOOP COUNTER
03A9	3450	1119	ARITLP: CALL REDENS ;THIS READS USING THE R0 POINTER
03AE	AD	1120	MOV R5,R ;SAVE TEMP IN R5
03AC	343F	1121	CALL REDSM1 ;THIS READS USING THE R1 PTR
03AE	B5BA	1122	JF0 ARADD ;JUMP IF ADDING
03B0	A5	1123	CLR F1 ;SUBTRACTING, SAVE CARRY
03B1	E054	1124	JNC R2 ;IF NO CARRY, SET F1
03B3	B5	1125	CPL F1
03B4	37	1126	AP3: CPL A ;GET 9'S COMPLEMENT
03B5	030A	1127	ADD R,#0100 ;ALWAYS SETS CARRY
03B7	76BA	1128	JF1 ARADD ;IF CARRY WS SET, KEEP IT
03B9	A7	1129	CPL C
03BA	4290	1130	ARADD: ORL R,#008H ;SO THAT DA WILL SET CARRY IF CARRY
		1131	;OUT OF NIBBLE
03BC	7D	1132	ADDC R,R5 ;DO ADD
03BD	57	1133	DA R ;DO DECIMAL ADJUST
03BE	3400	1134	ARITLP: CALL WRTEM1 ;WRITE OUT
03C0	EEA9	1135	DJNZ R6,ARITLP ;DONE
03C2	95	1136	CPL F0 ;RESTORE F0
03C3	83	1137	RET
		1138	*****
		1139	;
		1140	DSENP6—THIS S/R MOVES F HEX. TO PORT 6 OF THE DISPLAY 8279.
		1141	IT IS ALSO USED TO SET UP PORT 2 FOR THAT 8279
		1142	;
		1143	*****
		1144	;
03C4	23FF	1145	DSENP6: MOV R,#NONE
03C6	9A40	1146	PSNOVD: R#L P2,#SOLEH ;THIS ENTRY POINT ALLOWS ANY
03C8	8A10	1147	ORL P2,#PROG2 ;VALUE TO BE MOVD'D TO PORT 6
03CA	3E	1148	MOVD P6,R
03CB	83	1149	RET
		1150	*****
		1151	;
		1152	REGRED—THIS S/R SETS UP A TASK (IN ACCUM) IN BAKSTA
		1153	;
		1154	*****

LOC	OBJ	LINE	SOURCE STATEMENT
		1155	;
0300	2310	1156	REGRED: MOV A, #SWTSK ;ENTRY FOR REGISTER REQUEST
0300	0902	1157	BAKRED: MOV RL, #BAKSTR ;ENTRY FOR OTHER REQUESTS
0300	65	1158	STOP TCHT
0301	41	1159	ORL A, R01 ;GET BAKSTR
0302	R1	1160	MOV R01, A ;STORE BACK
0303	55	1161	STRT T
0304	83	1162	RET
		1163	*****
		1164	;
		1165	ENAFRG—S/R TO REENABLE THE FOREGROUND AFTER USING DISFRG
		1166	;
		1167	*****
		1168	;
0305	65	1169	ENAFRG: STOP TCHT
0305	B81A	1170	MOV R0, #R2FORG ;FOR INDIC ACCESS OF R2'
0308	F0	1171	MOV A, R08 ;GET READY TO REENABLE THE FOREGROUND
0309	537F	1172	ANL A, #07FH ;GET RID OF MSB
0308	2405	1173	JMP DSFRG1
		1174	*****
		1175	;
		1176	FAULTC—ROUTINE TO SEE IF HOME FAULT OR CLUTCH HALT
		1177	;
		1178	*****
		1179	;
		1180	FAULTC: ;SET UP TO R7' COMPLEMENTED
03D0	92EC	1181	JB4 FAULT1 ;JUMP IF NOT HOME TO SOFT FAULT
03DF	2383	1182	MOV A, #083H ;COME HERE FOR HOME FAULT
		1183	FAULT0: ;IN CASE POWER LOSS
03E1	2A	1184	XCH A, R2 ;SEE IF POWER LOSS
03E2	F2EC	1185	JB7 FAULT1 ;NO FAULT IF YES
03E4	2A	1186	XCH A, R2 ;ELSE GO ON AND DO FAULT STUFF
03E5	35	1187	DIS TCHT1 ;COME HERE IF FAULT
03E6	C6EC	1188	JZ FAULT1 ;IF ZERO FAULT, DON'T WRITE OUT
03E8	B82D	1189	MOV R0, #BFALT ;
03EA	3408	1190	CALL WRTRMB
03FC	24FA	1191	FAULT1: CALL T0FFSS ;TURN OFF SOLEN
03FE	64EE	1192	JMP \$
		1193	*****
		1194	;
		1195	IOWB—PART OF I/O ROUTINE, THIS WRITES TO BAM IFF TEST=1
		1196	ELSE ERROR IT WRITE TO BAM 1 IF BIT 4 OF DATA WORD
		1197	IS A . ELSE IT WRITE TO BOTH BAMS AFTERWARDS IT JUMPS
		1198	TO IOTSKS WHICH INCREMENTS THE I/O POINTER
		1199	;
		1200	*****
		1201	;
03F0	F1	1202	IOWB: MOV A, R01 ;SET UP TO R7FORG
03F1	12F5	1203	JB0 IOWB1 ;JUMP IF TEST (LOADED IN IOTSK3) = 1
03F3	84CE	1204	JMP IOERR ;ELSE JUMP TO ERROR
03F5	FD	1205	IOWB1: MOV A, R5 ;ELSE GET DATA WORD
03F6	3408	1206	CALL WRTRMB ;WRITE TO BAM
03F8	92FD	1207	JB4 IOWB2 ;JUMP IF WRITE TO BAM1 ONLY
03FA	C8	1208	DEC R0 ;ELSE DEC WRITE POINTER
03FD	3408	1209	CALL WRTRMB ;END WRITE TO BAM1 AND 2

LOC	OBJ	LINE	SOURCE STATEMENT
0401	F400	1210	JMP J04F52 JIF 101513 ; J04F52: JIF TO INCREMENT POINTER
		1211	;
0402		1212	ORG 4604
		1213	;
		1214	*****
		1215	*****
		1216	*****
		1217	*****
		1218	*****
		1219	*****
		1220	*****
		1221	*****
		1222	*****
0403	F404	1222	DONE5: MOV R2, 0004H ; INIT R2
0404	0421	1223	CALL F15013 ; DOES INCR OF R2 AND MOV. TO P5
0405	0423	1224	CALL F15013 ; THEN F15013 1ST TAPE. THEN ENVELOPE
0406	0425	1225	CALL F15013 ; THEN REPEAT
0407	7404	1226	CALL DONE ; DOES FINAL DEBOUNCING. IF F1=1
0408	7411	1227	JF1 DONE5 ; MAKE CHANGE
0409	0426	1228	DONE5: MOV R2, 0005H ; DONE WITH 4040 SET UP FOR NINE CHAP
0410	0427	1229	MOV R0, 00007 ; R0: FIRST OF KEYS SWITCHES
0411	0407	1230	JMP ONE5H ; FIRE FAULT FF ONE SHOTS
0412	04	1231	DONE5: MOV R, P5 ; WAIT TO SEE IF CHANGE IN
0413	0408	1232	R, R ; TAPE OF ENVELOPE
0414	0410	1233	JZ DONE5 ; JUMP IF NONE
0415	2340	1234	MOV R, 0000H ; R0: THEN ENVELOPE TO B0151A
0416	7405	1235	CALL B0151A
0417	0409	1236	JIF DONE5
0418	7407	1237	CALL REG001 ; EITHER REPEAT CHS OR REG CHS
0419	0408	1238	JIF DONE5
		1239	*****
		1240	*****
		1241	*****
		1242	*****
		1243	*****
		1244	*****
		1245	*****
		1246	*****
		1247	*****
		1248	*****
		1249	*****
		1250	*****
		1251	*****
		1252	*****
		1253	*****
0421	1A	1254	F15013: INC R2
0422	FA	1255	MOV R, R2
0423	31	1256	MOV P5, R ; ENTRY POINT TO LOOK AT NEXT INPUT
		1257	; OF 74150
0424	27	1258	F15013: CLF R
0425	4608	1259	JNT1 F15013
0427	17	1260	INC R ; MAKE ACCUM (0) = T1
0428	1E	1261	F15013: XOR R, R4 ; DO XOR
0429	37	1262	CPL R ; TO HELP IN PROGRAMMING
0430	1220	1263	JMP F15013
0431	1E	1264	INC R ; R5(0) = 1 IF XOR OF R4(0)



LOC	OBJ	LINE	SOURCE STATEMENT
		1265	; AND T1 = 1
0420	FE	1266	FDEK1: MOV R, R6 ; DO RR OF R4 AND R6
042E	77	1267	RR A
042F	FE	1268	MOV R6, A
0438	FC	1269	MOV R, R4
0431	77	1270	RR A
0432	FC	1271	MOV R4, A
0433	83	1272	RET
		1273	;*****
		1274	;
		1275	REGDSP-A BACKGROUND TASK WHICH READS THE REGSN REG AND DISPLAYS
		1276	THE 9 CHAR DISP. IF P.O. MODE, AND BATCH COUNT OR BATCH TOTAL
		1277	DO KEYBOARD
		1278	;
		1279	;*****
		1280	;
		1281	;USES R5, R1, R0, R6, R7, AND R3 IS INITED TO KEYSTR
		1282	;
0434	14FC	1283	REGDSP: CALL DISFRG
0436	891F	1284	MOV R1, #R7FORG ;SEE IF ON METER BASE
0438	F1	1285	MOV R, R1
0439	723D	1286	JB3 RGD00A ;JUMP IF YES, ELSE BLANK 4 CHAR DISP
043B	9484	1287	CALL BLKKB ;BLANK 4 DIGIT DISPLAY
043D	E31F	1288	RGD00A: MOV R0, #01FH ;GET R0 READY TO WORK WITH 9CHAR DISP
043F	FE09	1289	MOV R6, #009H ;(ADDR-1) AND R6 READY AS CHAR CTR
0441	852E	1290	MOV R1, #REGSN ;GET REGSN
0443	F1	1291	MOV R, R1
0444	531F	1292	ANL R, #01FH ;AND OUT BITS NOT REGISTER
0446	C675	1293	JZ RGDSP4 ;IF ZERO (NO SWITCHES DOWN) JUMP
0448	AD	1294	MOV R5, A ;ELSE SAVE IN R5
0449	FB	1295	MOV R, R3
044A	9279	1296	JB4 RGDSP4 ;JUMP IF P.O. MODE
044C	FD	1297	MOV R, R5 ;GET REGSN AND CONVERT TO
044D	2F00	1298	RGD00E: MOV R7, #0 ;BINARY FOR ADDR OF REG
044F	1255	1299	RGD00D: JB8 RGDSP3 ;SEE IF THIS IS REG SELECTED
0451	77	1300	RR A ;NO, SO RR A
0452	1F	1301	INC R7 ;AND INCR CTR
0453	844F	1302	JMP RGD00D ;AND SEE IF THIS IS REG SELECTED
0455	FF	1303	RGDSP3: MOV R, R7 ;GET BINARY #
0456	47	1304	SWAP R ;PUT IN MSN FOR ADDR OF REG
0457	A9	1305	RGDSP1: MOV R1, A ;GET READY FOR WRDTP
0458	85	1306	CLR F8 ;FOR WRDTP
0459	FR	1307	MOV R, R3 ;IF /CENTS, NEED TO IP R1 BY 1
045A	D25D	1308	JB6 RGD003 ;JUMP IF FCENTS
045C	19	1309	INC R1 ;INCR R1
045D	14EF	1310	RGD003: CALL WRDTP
		1311	;
		1312	;NOW HAVE TO DELETE LEADING ZEROS AND PUT IN DECIMAL POINT
		1313	;R0 POINTS TO CHAR 28, R1 HAS THE CHAR
		1314	;COUNT, ADD OFCH AFTER CLEARING MSN AND THIS GIVES COUNT FOR
		1315	;CHECKING FOR LEADING ZEROS
		1316	;
045F	F9	1317	MOV R, R1
0460	538F	1318	ANL R, #CLRMSN
0462	03EC	1319	ADD R, #0FCH

LOC	OBJ	LINE	SOURCE STATEMENT
0454	FF	1320	MOV R7, A ; MOVE LOOP CTR TO R7
0455	FB	1321	RGD005: MOV R, R00 ; GET CHR
0456	D383	1322	XRL R, #ZERO9C ; SEE IF = ZERO
0458	955F	1323	JNZ RGD008 ; JUMP OUT OF LOOP IF NOT
0460	37	1324	CPL A ; ELSE COMPL A FOR A CHR WHICH IS BLANK
0463	AF	1325	MOV R00, A ; STORE BACK IN SEG DECODE ARRAY
0465	DE	1326	DEC R0 ; TO LOOK AT NEXT CHR
0460	EF65	1327	DJNZ R7, RGD005
		1328	
046F	DE	1329	RGD008: DEC R0 ; NOW USE THIS TO FIND WHERE
			; TO R0X DECIMAL POINT
0470	EF6F	1330	DJNZ R7, RGD008 ; WHEN DONE, R0 POINTS TO CHR
0472	FB	1331	MOV R, R00 ; GET CHR SEGMENTS
0473	B7	1332	DEC A ; ADD DECIMAL PT DISPLAY
0474	AF	1333	MOV R00, A
0475	23EF	1334	RGD0P4: MOV R, #HSWTSK ; GET READY FOR MAINP
0477	E476	1335	JMP MAINP
0479	9484	1336	RGD0P4: CALL BLK0B ; BLANK 4 CHR DISPLAY
047B	FI	1337	MOV R, R5 ; P.O. MODE, IF REGSN = BATCH, DO
047C	5307	1338	RAI R, #007H ; DISPLAY OF KEYBOARD REG
047E	964D	1339	JNZ RGD00E ; JUMP IF NOT
0480	2360	1340	MOV R, #KBKREG ; DISPLAY KEYBOARD REG
0482	8457	1341	JMP RGD0P1
0484	B530	1342	BLK0B: MOV R1, #030H ; GET READY TO CLEAR 4 CHR DISPLAY
0486	27	1343	CLR A
0487	B764	1344	MOV R7, #004H ; GET LOOP CTR READY
0489	AF	1345	RGD0E1: MOV R01, A ; ZERO TO 4 CHR DISPLAY = BLANK
0489	19	1346	INC R1
048B	EF09	1347	DJNZ R7, RGD00E1
0490	B3	1348	RET
		1349	;*****
		1350	;
		1351	DISK02: S/R WHICH DISPLAYS THE KEYBOARD REGISTER. IT MUST WORRY
		1352	ABOUT SUCH THINGS AS PLACEMENT OF DECIMAL POINTS,
		1353	DISPLAYING A SERIPL, ETC. IT CALLS
		1354	WRTDSP (WRITE DISPLAY) WHICH IS A S/R WHICH READS A REGISTER
		1355	(RAM) POINTED TO BY R1 AND WRITES TO RAM POINTED TO BY R0.
		1356	FB=1 SAYS DON'T COMPL SEGMENTS AND R7 NEEDS THE # OF CHARS
		1357	TO BE MOVED. THE CHARACTERS PLACED IN RAM ARE FIRST CONVERTED
		1358	TO 7 SEGMENT CODE FOR THE 9 CHR DISPLAY, HENCE THIS ROUTINE
		1359	MUST COMPLEMENT THEM ADDING DECIMAL POINT WHERE APPROP.
		1360	;
		1361	;*****
		1362	;
048E	B950	1363	DISK0R: MOV R1, #KBKREG ; PUT IN R1 THE ADDR OF THE KEYBD REG
0490	85	1364	CLR R0
0491	95	1365	CPL R0 ; MAKE FB=1 FOR LATER
0492	BE64	1366	MOV R6, #004H ; INIT # OF CHR CTR
0494	B02F	1367	MOV R0, #02FH ; R0 PTS TO BEGINNING OF 4 CHR DISREG(-1)
0496	FB	1368	MOV R, R3 ; LOOK AT WHETHER TO READ FROM KBREG(0)
		1369	; OR KBREG(1)
0497	D290	1370	JBS DSK0RB ; IF JUMP, FROM (0)
0499	19	1371	INC R1 ; IT'S /CENTS, READ FROM (1)
049A	145F	1372	DSK0RB: CALL WRTDSP ; MOVE FROM KEYBD REG TO 4 CHR DISPREG
		1373	; RETURNS WITH R0 PTING TO MSD OF
		1374	; 4 DIGIT DISPLAY



LOC	OBJ	LINE	SOURCE STATEMENT
045C	FB	1375	MOV R, R3 ;SEE IF FCENTS MODE
045D	D2R1	1376	JB6 DSKBR1 ;JUMP IF YES
045F	B5	1377	CLR FB ;FOR LATER TEST
04A0	C8	1378	DEC R0 ;WENT TO PUT DEC TO 2ND MSDIGIT
04A1	18	1379	DSKBR1: INC BR0 ;ADD DEC PT TO CORRECT CHAR
04A2	95	1380	CPL FB ;FOR EASE OF TESTING
04A3	B6A0	1381	JF0 DSKBRF ;IF FCENTS DON'T JUMP
04A5	FB	1382	MOV R, R3 ;SEE IF .00 DISPLAY
04A6	B2P9	1383	JB5 DSKBRS ;JUMP IF .00 DISPLAY
04A8	83	1384	RET
04A9	B838	1385	DSKBRS: MOV R0, #838H ;IT'S FCENTS * .00
04AB	B892	1386	MOV BR0, #SERIPH ;SO DISPLAY SERIPH
04AD	83	1387	DSKBRF: RET
		1388	;*****
		1389	;
		1390	; IOTSK3—PART OF THE IO ROUTINE, CONHERE WHEN HAVE RECEIVED A
		1391	DATA WORD FROM THE UPI. STORE IT IN R5, CHECK THAT R4
		1392	IS NOT ERROR COMMAND, THEN JUMP TO PROPER ROUTINE
		1393	;
		1394	;*****
		1395	;
04AE	AD	1396	IOTSK3: MOV R5, R ;STORE DATA IN R5
04AF	74C4	1397	CALL DSEMP6 ;TURN OFF ENAIO
04B1	FC	1398	MOV R, R4 ;SEE IF ERROR COMMAND
04B2	F200	1399	JB7 ITSK1J ;JUMP IF YES
04B4	5202	1400	JB2 IOARB1 ;ELSE JUMP IF IOARB
04B6	32BE	1401	JB1 IOTSK6 ;OR JUMP TO IOAKB
04B8	B937	1402	MOV RL, #IOPTR ;ELSE IT'S IOPTR
04BA	FD	1403	MOV R, R5 ;GET DATA WORD
04BB	R1	1404	BR1, R ;WRITE NEW POINTER
04BC	F4CD	1405	JMP IOEND2
04BE	FD	1406	IOTSK6: MOV R, R5 ;IT'S IOAKB, SEE IF # OR CLEAR
04BF	03F6	1407	ADD R, #0F6H
04C1	C6D6	1408	JZ IOAKB1 ;IF ZERO, IT'S CLEAR KEYBOARD
04C3	07	1409	DEC R
04C4	96C8	1410	JNZ IOAKB2 ;IF JUMP SHOULD BE #
04C6	447A	1411	JMP DEBAT1 ;ELSE IT'S CLEAR BATCH
04C8	C7	1412	IOAKB2: MOV R, PSW ;SEE IF LEGAL CHAR
04C9	D2CE	1413	JB6 IOERR ;IF AUX CARRY, NOT LEGAL CHAR
04CR	FD	1414	MOV R, R5 ;ELSE GO TO ADD TO KEYBOARD
		1415	;
		1416	JMP FINNCH ;ONLY NEED WITH KLUDGE 1/0
04DC	4471	1416	JMP FINN01 ;ELSE, GO TO FINN01
04DE	B1C8	1417	IOERR: MOV R4, #0C8H ;THIS IS FOR BAD CMD OR DATA
04D0	E498	1418	ITSK1J: JMP IOTSK8
04D2	B91F	1419	IOARB1: MOV RL, #R7FORG ;GET COPY OF TEST SIGNAL
04D4	64F0	1420	JMP IOARB
04D6	44E4	1421	IOAKB1: JMP FINCLKS
		1422	;*****
		1423	;
		1424	; CHKTOT—S/R WHICH CHECKS THAT DREG + AREG = TOTAL IT CHECKS THAT
		1425	THE TWO COPIES OF EACH ARE =, THEN MOVES DREG TO T0REG1A2,
		1426	TOTAL TO BTEMP1A2, AREG TO T0REG1A2 AND BTEMP (B0M1)
		1427	THEN IT ADDS T0REG1 AND BTEMP1 (COPY OF
		1428	AREG) TO BTEMP1 AND COMPARES BTEMP WHICH HAS A
		1429	COPY OF TOTAL IN B0M2 THIS LEAVES A COPY OF TOTAL IN BTEMP

LOC	OBJ	LINE	SOURCE STATEMENT
		1438 ;	(BOTH BAMS) FOR LATER USE
		1439 ;	
		1432 ;	*****
		1433 ;	
0408	B508	1434	CH.TOT: MOV RL, #B0REG
040A	7415	1435	CALL COMPB8
041C	B878	1436	MOV R8, #BTDPREG
04DE	347E	1437	CALL MOVRE8
04EP	B928	1438	MOV RL, #BTOTAL
04E2	7415	1439	CALL COMPB8
04E4	B898	1440	MOV R8, #BTDPF
04E6	347E	1441	CALL MOVRE8
04EE	B918	1442	MOV RL, #B8REG
04EA	7415	1443	CALL COMPB8
04EC	B888	1444	MOV R8, #BTAPREG
04EE	347E	1445	CALL MOVRE8
04F8	B898	1446	MOV R8, #BTDPF
04F2	3475	1447	CALL MOVRE8
04F4	B578	1448	MOV RL, #BTDREG
04F6	B898	1449	MOV R8, #BTDPF
04F8	85	1450	CLR F8 ;READY TO ADD TDREG AND TAREG INTO
04F9	74B2	1451	CALL PROBC ;BTDPF
04FB	B898	1452	MOV RL, #BTDPF
04FD	74B8	1453	CALL COMPB8 ;SEE IF EQUAL TO COPY OF TOTAL
04FF	B888	1454	MOV RL, #B0REG ;MOVE THE PERMMENT REGS TO TTEMP LOCATIONS
0501	B888	1455	MOV R8, #BTDPF
0503	3476	1456	CALL MOVRE8
0505	83	1457	RET
		1458 ;	*****
		1459 ;	
		1466 ;	XCHAPT--THIS SUBROUTINE XCHANGES R4 AND STEPIN(X) AND R5 AND
		1461 ;	STEPTK(X)
		1462 ;	
		1463 ;	*****
		1464 ;	
0506	FB	1465	XCHAPT: MOV R, R3 ;GET PRISTA -
0507	5383	1466	RLE R, #803H ;EXPECTS R3 IN A
0509	4338	1467	ORL R, #STEPIN ;FORM ADDR FOR STEPIN(X)
050B	88	1468	MOV R8, R
050C	433C	1469	ORL R, #STEPTK ;GET ADDR FOR STEPTK(X)
050E	89	1470	MOV RL, R
050F	FC	1471	MOV R, R4 ;GET R4 READY FOR XCH
0510	28	1472	XCH R, R8 ;DO IT
0511	8C	1473	MOV R4, R ;COMPLETE XCHG
0512	FD	1474	MOV R, R5
0513	21	1475	XCH R, R4
0514	8D	1476	MOV R5, R
0515	FB	1477	MOV R, R3 ;GET R3 BACK IN THE ACCUM
0516	83	1478	RET
		1479 ;	*****
		1480 ;	
		1481 ;	PRINT--ROUTINE WHICH INITIALIZES STEPPERS, POSITIONS THEM, UPDATES
		1482 ;	THE BAML AND GENERALLY CONTROLS THE PRINT CYCLE
		1483 ;	IT USES R3 TO STORE THE PRISTA (PRINTER STATUS)--WHICH
		1484 ;	STEPPER, ENV OR TAPE, WHAT PART OF TASK BEING DONE, ETC



LOC	OBJ	LINE	SOURCE STATEMENT
		1485 ;	IT STORES IN R4 STEPTK(X), LSN = FINAL CODE (FROM KEYBD REG)
		1486 ;	MSN = CURRENT COMMAND TO STEPPER (X)
		1487 ;	IT STORES IN R5 STEPTK(X), LSN HAS NEXT POSITION EXPECTED,
		1488 ;	AND MSN HAS INFO ABOUT IF DONE POSITIONING, STEPPING ON OR
		1489 ;	CCW AND HOW MANY TIMES TRIED TO POSITION AND IF EXPECTING
		1490 ;	A 1/2 STEP VERIFY (-1)
		1491 ;	
		1492 ;	*****
		1493 ;	
		1494 ;	
0517	FB	1495	PRINT: MOV R,R3 ;SET UP TO KEYSTR
0518	929E	1496	JB4 PTSCX ;IF P.O. MODE, SOFT FAULT
0519	470E	1497	ORL R,#00FH ;MAKE KEYBOARD FULL
051C	FB	1498	MOV R0,R ;STORE IN KEYSTR
051D	14FC	1499	CALL DISFRG
051F	B835	1500	MOV R0,#PR1STR ;GET PR1STR
0521	B938	1501	MOV RL,#B38H ;WANT TO SEE IF OUT OF MONEY
0523	F1	1502	MOV R,R1
0524	1278	1503	JB8 PRTEXV ;JUMP IF YES
052E	B91F	1504	MOV RL,#R7FORG ;FOR POSSIBLE USE IN PRINT1
0528	F8	1505	MOV R,R0
0529	FB	1506	MOV R3,R ;INTD R
052A	D287	1507	JB6 PRINT1 ;IF DONE POSITIONING, DON'T NEED TO
		1508	;BRING IN STEPTK(X) AND STEPTK(X)
052C	FA	1509	MOV R,R2 ;MUST MAKE SURE NOT CLUTCH OR HOME
052D	37	1510	CPL R ;WAKEUP
052E	3234	1511	JB1 PRINT3 ;JUMP IF NOT
0530	F1	1512	MOV R,R1 ;FOR FAULTC
0531	37	1513	CPL R ;FOR ERASE OF NEXT TEST
0532	64DD	1514	JMP FAULTC ;JUMP TO SEE HOME FAULT OR CLUTCH HALT
0534	2405	1515	PRINT3: CALL XCHAPT ;BRING IN STEPTK(X) TO R4 AND STEPTK
		1516	; (X) TO R5
0536	B274	1517	JB5 PTSC2 ;DO JUMPS TO APPROPRIATE PTSCS
0538	9248	1518	JB4 PTSC1
		1519	;
		1520	;THIS IS BEGINNING OF PRINT WHEN JUST GET STATE OF NEW PRINT TASK
		1521	;AND DO PRELIMINARY CHECKING
		1522	;
		1523	PTSC8: ;TURN OFF KEYBD AND I/O STUFF
053A	2301	1524	MOV R,R1NR1 ;THIS SAYS LOOK ONLY AT PRINTER
053C	74CE	1525	CALL BAKRED
053E	B82E	1526	MOV R0,#REGSW
0540	FA	1527	MOV R,R0
0541	5308	1528	ANL R,#B00H ;GET PR1STR INITLED WITH TAP/ENW
		1529	;AND REPEAT
0543	47	1530	SWAP R ;REPEAT IS BIT 3 AND ENW IS BIT 2
		1531	;THIS TAPE IS NOT BIT 2
0544	4B	1532	ORL R,R3 ;SAVE INITING BIT IF EXISTS
0545	R310	1533	PTSC8: ADD R,#010H ;CHANGE TO PRINTER TASK 1
0547	FB	1534	MOV R3,R
		1535	;
		1536	;PTSC1 GETS INITIAL POSITION OF STEPPERS STARTED
		1537	;
0548	34D1	1538	PTSC1: CALL SNO
054B	FB	1539	MOV R,R3 ;SEE IF DONE WITH ALL FOUR

1515-11 MCS-48/UP1-41 MACRO ASSEMBLER V3 0
PTE 925X METER. 4-10-79

PAGE 29

LOC	OBJ	LINE	SOURCE STATEMENT
0540	37	1540	CPL A
0540	1254	1541	JB0 PRTFAB ;NO
0540	3254	1542	JB1 PRTFAB ;NO
0550	FE	1543	MOV R,R3 ;YES, GO TO NEXT TASK
0551	0310	1544	ADD R,#010H
0552	FE	1545	MOV R3,A
		1546	;
		1547	;PRTFB IS THE END OF TASKS 0 THRU 3
		1548	;IT DOES THE XCHG OF R4 AND R5 AND STEPIN(X) AND STEPTK(X)
		1549	;GETS THE NEXT STEPPER INTO PRISTA
		1550	;
0554	B40C	1551	PRTFAB: CALL XCHP1
0556	3260	1552	JB1 PRTFB ;MUST SEE IF DOING LAST SINCE MUST
		1553	;INCR LAST STEPPER TO 0
0558	17	1554	PRTFB5: INC R ;R3 IS IN A
0559	B635	1555	PRTFB7: MOV R0,PRISTA
0558	FE	1556	PRTFB8: MOV RB,R ;STORE PRISTA BACK
055C	F260	1557	JB7 PRTFB ;JUMP IF INITING
055E	E479	1558	PRTFB9: JMF MAINE2
0560	1264	1559	PRTFB3: JB0 PRTFB4
0562	A458	1560	JMP PRTFB5
0564	53FC	1561	PRTFB4: ANL R,#0FCH ;TO START AGAIN WITH ZERO STEPPER
0566	A459	1562	JMF PRTFB7
0568	37	1563	PRTFB6: CPL A ;FOR EDGE OF TESTING
0569	D25E	1564	JB6 PRTFB ;JUMP IF NOT DONE STEPPING
056B	34F8	1565	PRTFB8: CALL TOFSS ;DOE INITING. TURN OFF STEPPERS
056D	AD	1566	MOV RB,R ;AND SOLEN(SETS WITH 0 IN A), MAKE PRISTA = 0
056E	44E4	1567	JMF FNCLK5 ;GO TO CLEAR KEYBOARD (AND MAKE KEYBD NOT FULL)
0570	Z39C	1568	PRTFB9: MOV R,#09CH ;TURN OFF ALL PRINTER BITS, SINCE OUT OF MONEY
0572	E476	1569	JMP MAINE
		1570	PTSK08: MOV R,#004H ;P.O. MODE AND TRYING TO PRINT
		1571	;
		1572	JMF FAULT0 ;ERROR
		1573	;
		1574	;THIS IS CONTINUATION OF FINDING PRESENT POSITION
		1575	;OR POSITIONING IF TASK 2
		1576	;IT LOOKS FOR A VALID VERIFY CODE AND ACCEPTS THAT AS PRESENT
		1577	;POSITION IF CAN'T FIND JUST CONTINUES TO TRY IF TASK 3
		1578	;AND CAN'T POSITION, FAULT, UNLESS INITING, AND THEN RETRY.
		1579	PTSK2: CALL VER ;THIS TURNS OFF
		1580	;TIMER WAKEUP BIT AND RESTARTS
		1581	;PRINTER TIMER COUNTER AND
		1582	;CHECKS POSITION AND SETS BITS 567
		1583	;IN STEPTK ACCORDING TO TABLE BELOW
0576	B680	1584	JB0 PTSK28 ;JUMP IF FOUND NEXT POSITION
0578	FD	1585	MOV R,R5 ;ELSE SEE IF 2ND TIME TRY
0579	D290	1586	JB6 PTSK32 ;JUMP IF YES
057B	4340	1587	ORL R,#STPTRY ;ELSE MAKE 2ND TIME TRY = 1
057D	AD	1588	MOV R5,A ;PUT BACK IN R5
057E	A454	1589	JMP PRTFB ;GO TO END OF POSITIONING
0580	FE	1590	PTSK28: MOV R,R3 ;SEE IF PTSK 2 OR 3
0581	9202	1591	JB4 PTSK38 ;JUMP IF 3
0583	37	1592	PTSK29: CPL A ;SEE IF LAST STEPPER
0584	1254	1593	JB0 PRTFB ;GO TO PRINT FIN IF NOT
0586	3254	1594	JB1 PRTFB

LOC	OBJ	LINE	SOURCE STATEMENT
		1595	;
		1596	; THIS LAST STEPPER IS POSITIONED, ARE THE OTHERS?
		1597	; FOLLOWING IS DECODE OF BITS 567 OF STEPTK
		1598	; 765 DESCRIPTION
		1599	; 000 JUST DONE NSTEP, OR NOT POSITIONING, OR JUST STARTING
		1600	; 001 DONE WITH A FULL STEP
		1601	; 010 DOING 2ND TIME TRY FOR FULL POSITION
		1602	; 011 DONE POSITIONING
		1603	; 100 DOING HALF STEP
		1604	; 101 NO MEANING
		1605	; 110 NO MEANING
		1606	; 111 NO MEANING
		1607	;
0508	B93E	1608	MOV R1, #STPTK2 ; INIT TO STEPTK #2
050A	BE03	1609	MOV R6, #003H ; ONLY HAVE TO CHECK 3
050C	F1	1610	PTSK21: MOV R, R01 ; GET PTSKTK(X)
050D	37	1611	CPL A ; FOR EASE OF TESTING
050E	B254	1612	JBS PRTEA0
0509	D254	1613	JBS PRTEA0 ; JUMP OUT IF NOT DONE
0502	C9	1614	PTK21: DEC R1 ; ELSE DECREMENT POINTER
0503	EEBC	1615	DJNZ R6, PTSK21 ; DONE ALL?
0505	B406	1616	CALL XCHGDIR ; YES, SEE IF INITING, OR WHICH TASK
0507	92B1	1617	JB4 PTSK3A ; TASK3, MUST CLEANUP
0509	C409	1618	JMP PTSK2P ; TASK2, JUMP
		1619	;
		1620	; THIS IS PRINTER TASK 3 CODE, MOST OF IT IS IN COMMON WITH TASK 2
		1621	;
050E	FB	1622	PTSK32: MOV R, R3 ; COULDN'T POSITION
050C	F200	1623	JB7 PTSK33 ; JUMP IF INITING
050E	64EC	1624	PTSK3X: JMP FAULT1 ; ELSE SOFT FAULT
050B	E437	1625	PTSK33: JMP INITPW ; TRY AGAIN INITING
		1626	MOV R, #CHGDIR ; TIRED 2 TIMES AND COULDN'T
		1627	XRL R, R5 ; STEP BACK AND RETRY
		1628	MOV R5, R ; THIS SWITCHES DIRECTION
		1629	CALL NSTEP
		1630	CALL NSTEP ; NSTEP RETURNS WITH R5 IN A
		1631	XRL R, #CHGDIR ; SWITCH BACK TO ORIGINAL DIR
		1632	MOV R5, R
		1633	PTSK3X: CALL SNO ; SEND OUT NEW COMMAND
		1634	JMP PRTEA0
0502	FD	1635	PTSK38: MOV R, R5 ; POSITIONED, ARE WE TO FINAL POSIT
0503	37	1636	CPL A ; FOR EASE OF TESTING BIT 5 & 6
0504	D200	1637	JB6 PTSK39
0505	B200	1638	JBS PTSK39 ; JUMP IF NO
0508	FB	1639	MOV R, R3 ; GET READY FOR PTSK29
0509	B403	1640	JMP PTSK29
050B	D474	1641	PTSK39: CALL NSTEP ; NO, GET NEXT COMMAND
050D	3401	1642	CALL SNO ; SEND OUT NEW COMMAND
050E	B454	1643	JMP PRTEA0
0501	27	1644	PTSK3A: CLR A ; SINCE TO FINAL POSITION
0502	3A	1645	OUTL P2, A ; TURN ON SOLENOID
0503	340C	1646	CALL TOPTIC ; TURNS OFF PRCTR AND TIMER WAKEUP
0505	C40A	1647	JMP PTK200 ; JUMP
		1648	;
		1649	; THIS IS AFTER POSITIONING, TASK 4 OR 5, FIRST CHECK THAT NO TIMER



LOC	OBJ	LINE	SOURCE STATEMENT
		1650	;WAKEUP (IF YES, HOME DIDN'T HAPPEN WITHIN 88 MSECS OF CLUTCH AND
		1651	;IS A FAULT). THEN CHECK NOT TAPE OR ENV WAKEUP (JUST THOSE SIGNALS
		1652	;GOING AWAY, NO PROBLEM). THEN TURN OFF PHOCL BIT. SEE IF HOME = 1
		1653	;IF NOT, SEE IF CLUTCH. IF YES, START TIMER AND RETURN TO MAIN
		1654	;IF NOT CLUTCH, IF NOT TASK 5, JUST RETURN TO MAIN. ELSE BACK TO HOME.
		1655	;UPDATE BATCH STUFF AND COUNT AND END PRINT CYCLE AND RETURN TO MAIN
		1656	;IF HOME HAS LEFT, TURN OFF THE STEPPERS AND CLEAR THE PRINTER TIMER
		1657	;UPDATE DR AND AR AND CHECK RESULTS OF MOVES, AND CHANGE TO TASK 5 AND
		1658	;RETURN TO MAIN
		1659	;
0527	FR	1660	PRINT1: MOV R2 ;GET R2
0528	B2EC	1661	JB5 CHFAUL ;JUMP IF TIMER WAKEUP
0529	D2M	1662	JB6 PTSK4E ;JUMP IF TAPE OR ENV WAKEUP
0530	53FD	1663	RL R,#PHOCL ;TURN OFF PHOCL BIT
0531	65	1664	STOP TONT ;WHILE CHANGING R2
0532	RA	1665	MOV R2,R
0533	55	1666	STRT T
0534	F1	1667	MOV R,R1 ;SET UP TO R7FORG
0535	92ER	1668	JR4 PTSK41 ;JUMP IF LEFT HOME
0536	B2ER	1669	JB5 PTSK4T ;JUMP IF CLUTCH
0537	FB	1670	MOV R,R3 ;SEE IF TASK 4 OR 5
0538	92OR	1671	JR4 PRINTR ;RETURN TO MAIN IF TASK 4
0539	E479	1672	PRINT7: JMF MAINR2 ;IT'S TASK 4
0540	B2OF	1673	JB5 PRINT9 ;IF TASK 7, DON'T TURN OFF STEPPERS
0541	34FE	1674	CALL TOFFSS ;SINCE ANOTHER ENV COMING
0542	C41F	1675	PRINT9: JMF PTSK51 ;GO TO END OF PRINT
		1676	;
		1677	;
		1678	;HAVE ENV/TAPE WAKEUP, IF NEW ENVELOPE, GO TO TASK 6 (IF NOW TASK 4)
		1679	;OR TASK 7 (IF NOW TASK 5), OTHER WISE JUST TAPE OR ENV GOING AWAY,
		1680	;TURN OF WAKEUP AND GO BACK TO MAIN
		1681	;
0543	B92E	1682	PTSK4E: MOV R,R6GSH
0544	F1	1683	MOV R,R1 ;GET TAPE AND ENV
0545	D2OR	1684	JB6 PTSK4F ;IT'S ENVELOPE, JUMP
0546	23BF	1685	PTSK4H: MOV R,#PTAEN ;TO TURN OF TAPE/ENV WAKEUP
0547	E476	1686	JMF MAIN
0548	FB	1687	PTSK4F: MOV R,R8 ;SET UP TO PRISTA
0549	432B	1688	RL R,#B2BH ;CHANGE TO TASK 6 OR 7
0550	RA	1689	MOV R,R ;PUT BACK INTO PRISTA
0551	A4DC	1690	JMF PTSK4H
		1691	;
		1692	;FOUND CLUTCH. IF DR NOT ALREADY UPDATED (PTASK = 5 OR 7)
		1693	;START TIMER FOR 1ST 112 MSECS
		1694	;
0552	FB	1695	PTSK4T: MOV R,R3 ;SEE IF ALREADY GOT HOME
0553	925E	1696	JB4 PRTRFA ;JUMP IF YES
0554	53FC	1697	RL R,#BFCH ;CLEAR OUT TWO STEPPER BITS FOR 224 MSEC TIMER
0555	RA	1698	MOV R,R ;STORE INTO PRISTA
0556	23FF	1699	MOV R,#BFH ;GIVE 100 MSECS FOR HOME
0557	34AC	1700	CALL TOPTIC ;CHANGE PRCTR
0558	A4FE	1701	JMF PTSK42
0559	FB	1702	CHFAUL: MOV R,R3 ;SEE IF 1ST TIME THROUGH (1ST 112 MS)
0560	12F4	1703	JB8 HFAUL ;JUMP IF THRU 2ND
0561	10	1704	INC R ;ELSE INCR PRISTA



LOC	OBJ	LINE	SOURCE STATEMENT
05F8	23FF	1705	MOV R, #00FFH ; AND START 2ND 112 MSEC WAIT
05F2	04F9	1706	JMP PTSK4P
05F4	2382	1707	HFAULT: MOV R, #002H ; THIS IS NO HOME FAULT
05FE	64E1	1708	JMP FAULT0
		1709	;
		1710	; THIS IS CODE FOR WHEN LEFT HOME OR GOT CLUTCH --
		1711	; UP DATE AR AND DR, THEN GO TO TASK 5 OR 7
		1712	;
05F8	27	1713	PTSK41: CLR A
05F9	34AC	1714	PTSK4P: CALL TOPTIC ; THIS CLEARS PRCTR AND DOES STRT 1
05FB	FB	1715	MOV R, R3 ; ALREADY DONE TASK 4?
05FC	925E	1716	JB4 PRTRFA ; JUMP IF YES
		1717	PTSK42: MOV R, R2 ; IF POWER LOSS, JUST STOP
		1718	;
05FE	D433	1719	CALL MVTDR ; MOVE NEW DR TO DR FROM TDR
0600	B900	1720	MOV RL, #BTAREG
0602	B810	1721	MOV R0, #BAREG
0604	347E	1722	CALL MOVRRB ; MOVE NEW AR TO AR FROM TAR
		1723	;
		1724	; NOW UPDATE COUNT, BATCH COUNT
		1725	; AND BATCH TOTAL
		1726	;
0606	85	1727	CLR FB
0607	B830	1728	MOV R0, #BBATL ; NOW ADD KB TO BATCH COUNT
0609	7400	1729	CALL ARKENC
060B	B900	1730	MOV RL, #BRGTHD ; THIS REGISTER HAS 1000 IN IT FOR
		1731	; ADDING 1 000 TO BATCH COUNT
060D	B840	1732	MOV R0, #BBACT
060F	7402	1733	CALL ARKNC
0611	B903	1734	MOV RL, #BRGONE ; THIS REGISTER HAS ONLY 1 IN IT
0613	B850	1735	MOV R0, #BCOUNT ; THIS IS ADDR OF THE COUNT REGISTER
0615	7402	1736	CALL ARKNC
0617	B930	1737	MOV RL, #BBATL ; COPY BOTH THE BATCH REGISTERS AND THE
0619	B830	1738	MOV R0, #BBATL ; COUNT REGISTER TO BAH 2
061B	3478	1739	CALL MOVRRB ; WRITE TO BOTH BAHs
061D	C4DA	1740	JMP PTK200 ; CHANGE TO TASK-5 (OR 7)
		1741	;
		1742	; THE END OF THE PRINTER TASK. ITS CLEARS PRSTA IF NOT DOING
		1743	; CONSECUTIVE ENVS. CHECKS IF HAVE
		1744	;
		1745	PTSK51: MOV R0, #PRTSTA
061F	FB	1746	MOV R, R3 ; CHECK TAPE AND REPEAT
0620	B220	1747	IR5 PTSK57 ; 1ST SEEING IF DOING CONSEC ENVS
0622	B000	1748	MOV @R0, #000H ; NO, CLEAR PRSTA
0624	5220	1749	JB2 PTSK55 ; IT'S ENVELOPE, JUMP
0626	7220	1750	JB3 PTSK55 ; IT'S TAPE AND REPEAT, JUMP
0628	44E4	1751	JMP FNCLK5 ; THIS CLEARS KEYBD REG AND REINITs
		1752	; TDREG AND TAREG
062A	534C	1753	PTSK57: MOV R, #040H ; KEEP TAPE/ENV. REPEAT AND TASK 4
062C	00	1754	MOV @R0, A ; STORE IN PRSTA
062D	B834	1755	PTSK55: MOV R0, #KEYSTA ; NEED TO FIX UP R3 WITH KEYSTA
062F	FA	1756	MOV R, @R0
0630	00	1757	MOV R3, R
0631	6451	1758	JMP AKREG4
		1759	*****

LOC .OBJ	LINE	SOURCE STATEMENT
	1768 ;	
	1761 ;	REGMOV—S/R WHICH MOV CONTENTS OF TEMP DREG TO PERMANENT
	1762 ;	DREG (BOTH BRMS)
	1763 ;	
	1764 ;	*****
	1765 ;	
0633 B97A	1766	MOV RL, #DREG
0635 B808	1767	MOV RL, #DREG
0637 247E	1768	JMP MOVREG
	1769 ;	
	1770 ;	*****
	1771 ;	
	1772 ;	VER—THIS S/R GETS THE VERIFY CODE IF THE VERIFY
	1773 ;	CODE MATCHES THE NEXT POSITION, THEN IT LOOKS TO SEE
	1774 ;	IF THE VERIFY CODE MATCHES THE FINAL CODE. IN EITHER
	1775 ;	CASE, IF SETS FB TO LET THE CALLING ROUTINE KNOW THAT
	1776 ;	THE VERIFY MATCHED. IF THE STEPPER IS POSITIONED TO
	1777 ;	THE FINAL CODE, THEN THE DONE BIT IS SET AS WELL.
	1778 ;	
	1779 ;	*****
	1780 ;	
0639 85	1781	VER: CLR FB ;FB=0 WILL SAY NO MATCH BETWEEN PRESENT
	1782	;(STEPTK(X)) AND VERIFY
063A 34AA	1783	CALL TOPTI ;THIS TURNS OFF THE TIMER TASK BIT
	1784	;AND RESTARTS THE PRINTER TIMEP
063C 74C4	1785	CALL DSEMP6 ;PUTS PORT 2 IN THE RIGHT STATE
063E FB	1786	MOV R, R3 ;SET UP VERIFY ADDR
063F 5383	1787	ANL R, #883H ;ONLY WANT STEPPER(X) INFO
0641 E7	1788	RL R
0642 E7	1789	RL R
0643 BF18	1790	MOV R7, #818H ;CLEAR R7 WHERE STORE VER POS
	1791	
0645 3C	1792	VERLP: MOV P4, R ;USE THE ONE BIT TO TELL WHEN DONE
0646 A8	1793	MOV R8, R ;SEND OUT ADDR TO VADR
		;SAVE IN R8
0647 364A	1794	JTB VER1 ;JUMP IF GET 0 BACK (INVERTED OUTPUT)
0649 1F	1795	INC R7
064A FF	1796	VER: MOV R, R7 ;DO RL OF R7
064B F253	1797	JB7 VER2 ;UNLESS DONE
064D E7	1798	RL R
064E FF	1799	MOV R7, R
064F FB	1800	MOV R, R8 ;GET BACK POINTER
0650 17	1801	INC R ;INCR IT
0651 C445	1802	JMP VERLP
0653 FD	1803	VER2: MOV R, R5 ;SEE IF LOOKING FOR -1 VERIFY
0654 F26F	1804	JB7 VER8 ;JMP IF YES
0656 DF	1805	XRL R, R7 ;NO COMPARE NEXT POSITION
	1806	;WITH VERIFY POSITION
0657 530F	1807	ANL R, #CLRASH
0659 C65C	1808	JZ VER5 ;JUMP IF MATCH
065B 83	1809	RET ;ELSE RET
065C FD	1810	VER5: MOV R, R5 ;STEP DONE, SET BIT
065D 539E	1811	ANL R, #DIDONEP ;REITER CLEARING TWO DONE BITS
065F 4328	1812	ORL R, #STPDON
0661 AD	1813	MOV R5, R
0662 95	1814	CPL FB ;FOUND A MATCH

LOC	OBJ	LINE	SOURCE STATEMENT
0653	FC	1815	MOV R,R4 ;SEE IF FINAL POSITION = VER POSIT
0664	DF	1816	XRL R,R7
0665	538F	1817	ANL R,#CLRMSH
0667	C66A	1818	JZ VER3
0669	B3	1819	PET ;RETURN IN NOT DONE POSITIONING
066A	FD	1820 VER3:	MOV R,R5 ;GET READY TO SET DONE POSITIONING
066E	476A	1821	ORL R,#POS2ON
066D	AD	1822	MOV R5,A
066E	B3	1823	PET
066F	529F	1824 VER8:	ANL R,#DONEP ;CLEAR TWO DONE BITS
0671	AD	1825	MOV R5,A
0672	95	1826	CPL FB ;DONE POSITIONING 1/2 STEP
0673	B3	1827	PET ;(PP02881V)
		1828 ;*****	
		1829 ;	
		1830 ;	NSTEP—THIS SUBROUTINE CHANGES THE STEPPER'S INSTRUCTION
		1831 ;	TO DO THE NEXT STEP. IT MUST TAKE INTO ACCOUNT THE
		1832 ;	DIRECTION TO CHANGE THE STEPPER INST (MSH STEP(X)).
		1833 ;	THEN IT SEES IF THE LAST TIME CHANGE WAS FOR A 1/2
		1834 ;	POSITION. IF YES, IT CHANGES THAT. IF NO, IT MUST INC
		1835 ;	OR DECR THE NEXT POSITION (LSN OF STEPT(X)).
		1836 ;	
		1837 ;*****	
		1838 ;	
0674	B5	1839 NSTEP:	CLR FB ;NOT INITING
0675	FD	1840 NSTEP1:	MOV R,R5 ;FIRST GET DIRECTION
0676	929D	1841	JB4 NSTEP8 ;JUMP IF INCREASE
0678	FC	1842	MOV R,R4 ;IT'S DECREASE, WILL DO A RR, IF B4=1
0679	9295	1843	JB4 NSTEP9 ;MUST JUMP
067B	53F8	1844	ANL R,#CLRLSH ;
067D	77	1845	RR R ;ELSE ROTATE RIGHT
067E	AF	1846 NSTEP8:	MOV R7,R ;SAVE IN R7
067F	FC	1847 NSTEP1:	MOV R,R4 ;NOW PUT NEW COMMAND BACK TO R4
0680	538F	1848	ANL R,#CLRMSH ;ZERO OUT MSH
0682	4F	1849	ORL R,R7 ;OR IN THE NEW COMMAND
0683	FC	1850	MOV R4,R ;PUT BACK
0684	B58C	1851	JEB NSTEP6 ;JUMP IF INITING
0686	FD	1852	MOV R,R5 ;GET R5 TO CHECK ON LAST TIME -1
0687	539F	1853	ANL R,#DONEP ;TURN OFF STEP AND DONE POSIT BITS
0689	F28F	1854	JBZ NSTEP3 ;WAS EXPECTED, JUMP IF YES
068B	4388	1855	ORL R,#STPM1 ;ELSE MAKE THE NEXTPOSITION -1
068D	AD	1856 NSTEP2:	MOV R5,A ;STORE BACK IN R5
068F	B3	1857 NSTEP:	PET ;AND RETURN
		1858 ;	
		1859 ;	NEED TO INCR OR DECR NEXT POSITION
		1860 ;	OR DEC IF (R5(4)=0)
		1861 ;	
068F	9296	1862 NSTEP3:	JB4 NSTEP5 ;B4=1 SAYS INCREASE
0691	B7	1863	DEC R ;THIS IS DECREASE
0692	537F	1864 NSTEP6:	ANL R,#STPM1 ;NOW MAKE NEXT TIME -1 = 0
0694	C48D	1865	JMP NSTEP2
0696	17	1866 NSTEP5:	INC R ;ELSE INCREASE R
0697	C492	1867	JMP NSTEP6
0699	BF18	1868 NSTEP7:	MOV R7,#010H ;FOR RL WITH MSB = 1
069B	C47E	1869	JMP NSTEP1



LOC	OBJ	LINE	SOURCE STATEMENT
0690	FC	1870	NSTEP8: MOV R, R4 ; THIS IS FOR INCREASE
069E	F299	1871	JB7 NSTEP7 ; JUMP IF RL GOES TO LSB
06A0	53F6	1872	ANL R, #CLR15H
06A2	F7	1873	R, R ; INCREASE MEANS PL
06A3	C47E	1874	JMP NSTEP8
06A5	BF0E	1875	NSTEP9: MOV R7, #080H ; FOR RR WITH LSB = 1
06A7	C47E	1876	JMP NSTEP1
		1877	*****
		1878	;
		1879	PTSK29--PART OF PRINT ROUTINE
		1880	;
		1881	*****
		1882	;
		1883	; MUST GET KEYBOARD REG INTO FINAL POSITION(X) AND SET UP DIRECTION
		1884	; WHICH DEPENDS ON DIFFERENCE BETWEEN FINAL AND PRESENT POSITION
		1885	;
06A9	BF04	1886	PTSK2F: MOV R7, #084H ; GET READY TO MOVE KEYBD REG TO
		1887	; STEPIN(X)
06AE	B96E	1888	MOV PL, #KBKREG
06AD	B834	1889	MOV R8, #KEYSTA ; SEE IF FCNTS OR NOT
06AF	F8	1890	MOV R, #R8
06B0	D2B3	1891	JB6 PTK204 ; JUMP IF FCNTS
06B2	19	1892	INC R1 ; ELSE READ FROM KBREG(1)
06B3	B836	1893	PTK204: MOV R8, #STEPIN
06B5	343F	1894	PTSK25: CALL REDEMI
06B7	38	1895	XCHD R, #R8 ; MOVE KEYBD(X) TO STEPIN(Y)
06B8	18	1896	INC R8
06B9	EFB5	1897	DJNZ R7, PTK25 ; DONE?
06BB	B804	1898	MOV R3, #084H ; YES, NOW PICK INCR OR DEC
		1899	; NOTE USING R3(ALL OTHER REGS
		1900	; BUSY), WILL REESTABLISH AT END
06BD	B4B5	1901	PTSK26: CALL XCHAPT ; GET STEPIN(X) AND STEPTK(X)
06BF	FD	1902	MOV R, R5 ; GET STEPTK(X)
06C0	530F	1903	ANL R, #CLR15H ; - PRESENT (STEPTK(X))
06C2	AE	1904	MOV R6, R ; SAVE TEMPORARILY IN R6
06C3	FC	1905	MOV R, R4 ; GET STEPIN(X)
06C4	530F	1906	ANL R, #CLR15H
06C6	37	1907	CPL R
06C7	6E	1908	ADD R, R6 ; THIS IS COMPLEMENT OF RESULT
06C8	F2C0	1909	JB7 PTK200 ; SO IF JUMP, RESULT (+) NEED TO INC
06CA	FE	1910	MOV R, R6 ; GET STEPTK(X) WITH MSN = 0
06CB	C4D3	1911	JMP PTK201
06CD	37	1912	PTK200: CPL R ; SEE IF NO CHANGE (RESULT COMPL = 0)
06CE	C6D4	1913	JZ PTK202 ; JUMP IF NO CHANGE
06D0	FE	1914	MOV R, R6 ; THIS IS INCREASE
		1915	; R6 IS STEPTK(X) WITH MSN = 0
06D1	4318	1916	ORL R, #010H ; SO INCREASE
06D3	AD	1917	PTK201: MOV R5, R ; PUT STEPTK IN R5 FOR NSTEP
06D4	B4B5	1918	PTK202: CALL XCHAPT ; RESTORE STEPIN(X) AND STEPTK(X)
06D6	1B	1919	INC R3 ; INCR POINTER
06D7	FB	1920	MOV R, R3 ; SEE IF DONE YET
06D8	52B0	1921	JB2 PTK26 ; JUMP IF NOT
06DA	B835	1922	PTK200: MOV R8, #PRISTA ; REESTABLISH PRISTA IN R
06DC	FB	1923	MOV R, #R8 ; FOR PRIFB3
06DE	B318	1924	ADD R, #010H ; GOES TO TASK 3



LOC	OBJ	LINE	SOURCE STATEMENT
B3DF	R45B	1925	JMP PRTEF8 ;END JUMP
		1926	;
		1927	*****
		1928	;
		1929	;
		1930	INIT -- THIS IS INITIALIZATION FOR THE FRIDEN ELECTRONIC POSTAL
		1931	METER MICRO CODE IT INIT'S VARIOUS REGISTERS.
		1932	;
		1933	;
		1934	;
		1935	;
		1936	;
		1937	;
		1938	;
		1939	;
		1940	;
		1941	;
		1942	;
		1943	;
		1944	*****
		1945	;
		1946	;
		1947	;
B5E1	Z34E	1948	INIT: MOV R, #SOLEND ;FIRST TURN OFF SOLENOID
B5E2	3P	1949	OUT P2, R
B5E4	27	1950	CLR R ;NOW CLEAR OUT ALL RAM
B5E5	B83F	1951	MOV R0, #03FH
B5E7	AB	1952	INIT1: MOV R0, R
B5E8	E8E7	1953	DJNZ R0, INIT1
B5EA	D5	1954	SEL R01 ;FIRST FIX UP FOREGROUND REGISTERS
B5EP	B92A	1955	MOV P1, #SPHACC ;INI FOR CALLS TO INTERRUPT ROUTINES
B5ED	B82A	1956	MOV R0, #REG87 ;INI FOR START OF DISPLAY & DEBOUNCING
B5EF	BAA8	1957	MOV R2, #0FAH ;INI CHARACTER POINTER TO NOT DISPLAY
B5F1	C5	1958	SEL R00 ;GO TO BANK 0 REGISTERS
B5F2	B536	1959	MOV RL, #PRCTR ;GET ADDR OF PRCTR TO INITIALIZE IT
B5F4	B1FF	1960	MOV R01, #MINONE
B5F6	Z3ER	1961	MOV R, #TIMCNT ;TO 255*400 MICROSECS OR APPROX
		1962	;
		1963	;
B5F8	62	1964	MOV T, R ;100 MILLISECS OF WAIT UNTIL START TO
B5F9	25	1965	EN TONT1 ;ACCESS RAM, B2435, ETC.
		1966	;
		1967	;
		1968	;
B5FA	55	1969	STRT T ;THAT SYSCLR IS GONE (HOPEFULLY)
		1970	;
B5FB	FR	1971	WAIT: MOV R, R2 ;START THE TIMER
B5FC	C6FB	1972	JZ WAIT ;WHEN FOREGROUND COUNTS DOWN
		1973	;
B5FE	05	1974	EN I ;PRCTR IT WILL SET A BIT IN BAKSTR
		1975	;
B5FF	R92D	1976	MOV R1, #REBLT ;ENABLE THE POWERLOSS INTERRUPT
B7B1	343C	1977	CALL RED8MB ;SEE IF METER FAULTED
B7B3	65	1978	STOP TONT ;IF NOT ZERO, FAULT, SHOULDN'T EVEN
B7B4	96B4	1979	INZ \$;BE EXECUTING, JUST STOP WITH TONT OFF

LOC	OBJ	LINE	SOURCE STATEMENT			
0706	55	1980	STRT	T	;FAULT DOES THAT	
0707	740E	1981	CALL	COMPBN	;R1 POINTS TO BNAWPO, SEE IF BOTH =	
0709	AC	1982	MOV	R4, A	;GETTING METTYPE READY	
0708	740E	1983	CALL	COMPBN	;GETTING METTYPE BITS	
070C	65	1984	STOP	TDNT	;MAKE SURE NOT ZERO	
070D	C68D	1985	JZ	\$;JUMP IF YES	
070F	55	1986	STRT	T		
0710	B904	1987	MOV	R1, #004H	;GET READY TO MOVE BNAWPO TO LSH	
0712	31	1988	XCHG	R, R1		
0713	B917	1989	MOV	R1, #METYPE		
0715	AL	1990	MOV	R1, A	;STORE INTO METTYPE	
071E	534E	1991	RAL	R, #040H	;NOW GET FCENTS	
0718	B32C	1992	MOV	R1, #REG00	;FIRST STORE INTO REG00	
071A	AL	1993	MOV	R1, A		
071B	430A	1994	ORL	R, #006H	;OR WITH NO KEY PUSHED	
071D	B934	1995	MOV	R1, #KEYSTA		
071F	AL	1996	MOV	R1, A	;STORE IN KEYSTA	
		1997	;NOW CHECK BNA CONTENTS FOR EQUAL CONTENTS IN EACH REGISTER.			
		1998	;			
0720	345E	1999	CALL	CLRKYR	;CHECKS THAT TWO COPIES OF DR	
		2000	;ARE =, TWO COPIES OF AREG AND			
		2001	;TOTAL ARE LIKEWISE =, THEN MOVES			
		2002	;DREG TO TDREG, AREG TO TAREG			
		2003	;TOTAL TO TEMP AND CHECKS THAT TDREG			
		2004	;+ TAREG = COPY OF TOTAL			
0722	B9A2	2005	MOV	R1, #BRGTH0	;SEE IF COUNT INCR REGISTERS ARE OK	
0724	BEB0	2006	MOV	R6, #000H	;CHECK MORE THAN A REGISTERS WORTH	
0726	741E	2007	CALL	CHFL0T	;RETURNS WITH 0 IF NO ERROR	
0728	AA	2008	MOV	R2, A	;CLEAR OUT R2	
0729	B83E	2009	MOV	R0, #PRCTR	;NOW LET FOREGROUND RUN FOR 100 MSECS	
072B	B87F	2010	MOV	R0, #0FFH		
072D	74D5	2011	CALL	ENAFRG	;ENABLE TO ALLOW SETTING OF REGSN	
		2012	;AND P.O. MODE, NORM, ETC.			
072F	FA	2013	WAIT1:	MOV	R, R2	
			;WAIT FOR FOREGROUND TO MAKE NON ZERO			
0730	37	2014	CPL	A	;FOR EASE OF TESTING IF 100 MSEC	
0731	B22F	2015	JBS	WAIT1	;HAS COME UP (8 PASSES OF FOREGND)	
0733	14FC	2016	CALL	DISFRG	;TURN OFF DISPLAY WHILE DOING NEXT	
0735	B810	2017	MOV	R2, #SWTSK	;CLEAR TIMER BIT AND LEAVE REGTSK	
0737	B81F	2018	INITPX:	MOV	R0, #R7F0RG	;SEE IF CONNECTED TO A METER BASE
0739	FE	2019	MOV	R, AREG		
073A	723E	2020	JBS	INITPX	;JUMP IF YES	
073C	44E4	2021	JMP	FNCLY5	;THIS DOES CLEAR KBREG AND	
		2022	;GOES TO MAIN			
073E	923E	2023	INITPX:	JBS	\$;IF NOT AT HOME, JUST STOP
0740	B240	2024	JBS	\$;IF CLUTCH PENDING, JUST STOP	
0742	B935	2025	MOV	R1, #PPTSTA	;GET READY TO INIT PRINTER STAT	
0744	B100	2026	MOV	R1, #000H	;THIS TELLS PRINTER ROUTINE INITING	
0746	B029	2027	MOV	R4, #009H	;GET READY TO STEP PRINTER TO 9 BY	
0748	B008	2028	MOV	R3, #000H	;FORCE, WILL SEND 30 INCR STEPS	
074A	B010	2029	MOV	R5, #010H		
074C	34D1	2030	INITP0:	CALL	SND	;SEND STEP COMMAND AND RESTART TIMER
074E	16	2031	INC	R3	;INCREMENT THE STEPPER POINTER	
074F	FE	2032	MOV	R, R3	;SEE IF HAVE DONE 4	
0750	37	2033	CPL	A	;FOR EASE OF TESTING	
0751	125C	2034	JBS	INITP2		

BAD ORIGINAL

LOC	OBJ	LINE	SOURCE STATEMENT
0753	325C	2035	JB1 INITP2
0755	37	2036	CPL A
0756	F261	2037	JB7 INITPF ;SEE IF HAVE SENT 32 COMMANDS
0758	85	2038	CLR FB
0759	95	2039	CPL FB ;THIS IS FOR USE BY NSTEP1
075A	D475	2040	CALL NSTEP1 ;GET NEXT COMMAND
075C	FA	2041	INITP2: MOV A,R2 ;GET COPY OF R2 TO SEE IF TIMER
075D	E24C	2042	JB5 INITP8 ;HRS TIMED OUT
075F	E45C	2043	JMP INITP2 ;ELSE LOOP
0761	B837	2044	INITPF: MOV R0,#STEPIN-1 ;INI THE INSTRUCTION TO THE STEPPERS
		2045	;INI R0 TO STEPIN(0)
0763	FC	2046	MOV A,R4 ;THIS SERVES TO INI BOTH
		2047	;STEPTK AND STEPIN
		2048	
0764	BF08	2049	MOV R7,#000H ;INI A LOOP COUNTER
0766	18	2050	INIT4: INC R0
0767	AB	2051	MOV @R0,A ;MOVE THE COMMAND TO STEPIN(X) AND STEPTK(X)
0768	EF66	2052	DJNZ R7,INIT4 ;IT IS 029H
076A	P917	2053	MOV P1,#METYPE ;WANT TO SEE IF 3 OR 4 STEPPER METER
076C	F1	2054	MOV A,@R1
076D	F274	2055	JB7 INITF ;JUMP IF 4
076F	27	2056	CLR A ;ELSE MUST CLEAR PRESENT AND FINAL POSITION
0770	AB	2057	MOV @R0,A
0771	B838	2058	MOV R0,#STEPIN+3
0773	AB	2059	MOV @R0,A
0774	44E4	2060	INITF: JMP FNCLK5 ;CLEARS KEYBOARD AND DISPLAYS AND
		2061	;GOES TO MAIN
		2062	;
0776	65	2063	MAINR: STOP TCHT ;STOP TCHT WHILE CHANGING BAKSTA
0777	5A	2064	ANL A,R2 ;A HAS MASK IN IT TO DROP A BAKSTA BIT
0778	AB	2065	MOV R2,A ;PUT BACK IN BAKSTA
0779	74D5	2066	MAINR2: CALL ENAFRG
077B	B834	2067	MOV R0,#KEYSTA ;GET KEYSTA TO R3 IN CASE KEYBD
077D	F8	2068	MOV A,@R0 ;OR REGDSP
077E	AB	2069	MOV R3,A
		2070	;
		2071	;THIS IS THE MAIN BACKGROUND LOOP WHICH DETERMINES WHICH BACK
		2072	;GROUND TASK SHOULD BE JUMPED TO IF ANY.
		2073	;
077F	FA	2074	MAIN: MOV A,R2 ;GET NEW COPY OF BAKSTA
0780	F280	2075	JB7 \$
0782	9292	2076	JB4 REGDSJ
0784	1288	2077	JBA MAIN2 ;IF BIT A ON, NO KEYBD, I/O
0786	5294	2078	JB2 IORTSK
0788	7290	2079	JB3 KEYBDJ
078A	5362	2080	MAIN2: ANL A,#062H ;ISOLATE THE PRINTER TASKS
078C	C67F	2081	JZ MAIN
078E	A417	2082	JMP PRINT
0790	4484	2083	KEYBDJ: JMP KEYBD
0792	8434	2084	REGDSJ: JMP REGDSP
		2085	*****
		2086	;
		2087	IORTSK--THE ROUTINE WHICH SERVICES I/O REQUESTS. THE METER IS THE
		2088	MASTER IN CONTROLLING A I/O TRANSFERS TO AND FROM THE
		2089	UPI. EXCEPT THAT THE UPI TELLS THE METER SOMETHING IS TO

LOC	OBJ	LINE	SOURCE STATEMENT
		2898 ;	BE DONE BY PULLING /EXTINTS LOW. WHICH CAUSES A I/O WAKEUP.
		2899 ;	THEN THE IOTSK ROUTINE READS UP1 STATUS AND SENDS A COMMAND
		2892 ;	TO LOAD THE UP1'S DBB. THE IOTSK AGAIN READS THE STATUS UNTIL
		2893 ;	THE UP1 HAS DONE THAT, THEN READS THE COMMAND, THEN AFTER
		2894 ;	DOING WHAT IS NECESSARY AND THEN READING STATUS AGAIN. EITHER
		2895 ;	THE DATA WORD IS READ, OR IT IS SEND TO THE UP1. IF IT WAS A
		2896 ;	DATA WORD TO THE METER, THE METER IS DIRECTED TO DO WHATEVER
		2897 ;	IS NECESSARY WITH THE COMMAND AND DATA. THE COMMANDS WHICH
		2898 ;	THE METER IS ABLE TO DO ARE: 0-ILLEGAL. 1-COPY DATA WORD TO
		2899 ;	IOPTR. 2-ADD CHRG TO KEYBD REG (OR CLR KEYBOARD OR BATCH).
		2100 ;	3-READ BMM USING THE IOPTR (AND THEN INCR THE IOPTR).
		2101 ;	4-WRITE BMM USING THE IOPTR (AND THEN INCR THE IOPTR).
		2102 ;	THE LAST COMMAND IS ONLY POSSIBLE IF THE SIGNAL /TEST IS TRUE
		2103 ;	WHICH IS POSSIBLE ONLY WHEN THE METER IS UNCOVERED AT THE
		2104 ;	FACTORY.
		2105 ;	
		2106 ;	*****
		2107 ;	
0794	14FC	2108	IOTSK: CPL DISERG
0796	27	2109	IOTSK2: CLR R
0797	AC	2110	MOV R4, R
0798	85	2111	IOTSK0: CLR F0 ; CLEAR THE CURRENT COMMAND REGISTER
		2112	; F0=1 SAYS COMMAND TO LOAD OR READ DBB
0799	55	2113	IOTSK1: STRT T ; HAS BEEN SENT
079A	E537	2114	MOV RL, #IOPTR ; FOR LOOPING
079C	F1	2115	MOV R, R01 ; FOR LATER USE
079D	AB	2116	MOV R0, R
079E	AB	2117	MOV RL, R ; FOR THE READ AND WRITE BMM IOTSKS
079F	65	2118	STOP TCNT ; TO KEEP FOREG FROM CHANGING P6
07A0	2309	2119	MOV R, #ENAVIO ; THIS WILL FIX-UP PORT 6 OF THE DISPLAY
07A2	7406	2120	CALL P6MOV0 ; B279 TO ENABLE THE I/O
07A4	8A30	2121	ORL P2, #DPROG ; THIS TURNS OFF BMM ENABLES TO BUSS
07A5	E901	2122	IOTSK5: ORL PL, #001H ; READY TO READ UP1 STATUS
07A8	08	2123	INR R, BUS ; READ STATUS
07A9	3299	2124	JB1 IOTSK1 ; END IF INPUT BUFFER FULL
07AB	5299	2125	JB2 IOTSK1 ; END IF UP1 BUSY
07AD	AS	2126	CLR F1
07AE	72B1	2127	JB3 IOTSK0
07B0	B5	2128	CPL F1 ; F1=1 SAYS DATA IS COMING (VS COMMAND)
07B1	12D1	2129	IOTSK0: JBR IOTSK0 ; JUMP TO IOTSK0 IF OUTPUT BUFFER FULL
07B3	B60E	2130	JF0 IOTSKC ; JUMP IF ALREADY SENT COMM TO UP1
07B5	4330	2131	ORL R, #030H ; SEND BACK UP1 STAT ORED WITH 030H
07B7	91	2132	MOVX R01, R ; SEND COMMAND (RL IS PHONEY ADDR)
07B8	95	2133	CPL F0 ; SAYS COMMAND SENT
07B9	E499	2134	JMP IOTSK1
07BB	FC	2135	IOTSKC: MOV R, R4 ; NO, WHAT IS COMMAND
07BC	F2C5	2136	JB7 IOTSKC ; IF THIS BIT SET, HAD IO ERROR
07BE	D303	2137	XRL R, #IORD0 ; SEE IF IT WAS IO READ (FROM BMM)
07C0	96D0	2138	JNZ IODERR ; JUMP IF NOT (MUST BE)
07C2	9900	2139	PNL PL, #000H ; GOING TO SEND DATA TO UP1, CLR P10
07C4	FD	2140	MOV R, R5 ; THIS IS BMM DATA
07C5	91	2141	IOTSKC: MOVX R01, R ; ADDR IN R1 IS PHONEY, NEED /R
		2142	; IF JUMP TO HERE, HAD ERROR, P10=L, R
		2143	; HAS ERROR COMMAND
07C6	B21F	2144	IOEND: MOV RL, #RZF0RG ; IF TEST BIT ON, STAY IN I/O

LOC (CJ)	LINE	SOURCE STATEMENT	
07C8 F1	2145	MOV	R, R01
07C9 1296	2146	JBB	IOTSK2 ; JUMP IF YES
07C8 74C4	2147	CALL	DSENP6 ; TURN OFF ENA10
07C0 23FB	2148	IOEND2: MOV	R, #IOTSK ; GET READY TO TURN OFF WAKEUP BIT
07CF E476	2149	JMP	MAINR
07D1 9500	2150	IOTSKD: RHL	PL, #000H ; OUTPUT BUFFER FULL, 1ST TURN OFF P18
07D3 08	2151	INS	R, BUS ; READ COMMAND/DATA
07D4 55	2152	STRT	T ; DONE WITH P6, RESTART T
07D5 76F0	2153	JF1	ITSK3J ; JUMP IF DATA
07D7 AC	2154	MOV	R4, R ; IT'S COM1, STORE IN R4
07D8 07	2155	DEC	R ; CHECK TO SEE IF LEGAL COMMAND
07D9 53FC	2156	RHL	R, #0FCH
07DB 09F	2157	JZ	ITSK01 ; JUMP IF LEGAL
07DD 84CE	2158	IOERRM: JMP	IOERR
07DF FC	2159	ITSK01: MOV	R, R4 ; IT'S LEGAL, SEE IF IOR08
07EB D383	2160	XEL	R, #IOR08
07E2 9698	2161	JNZ	IOTSK8 ; JUMP TO GET DATA WORD IF NOT IOR08
07E4 74C4	2162	CALL	DSENP6 ; MUST DO THIS BEFORE READING BAH
07EE 343C	2163	CALL	RED08B ; IT'S IOR08, ZERO WORD
07EB AD	2164	MOV	R5, R ; STORE IN R5
07E9 B937	2165	IOTSK5: MOV	PL, #IOPTR ; GET READY TO INCR IOPTR
07EB 11	2166	INC	R01 ; DO IT
07EC 76D6	2167	JF1	IOEND ; JUMP HERE IF END OF IOR08
07EE E495	2168	JMP	IOTSK2
07F0 84FE	2169	ITSK3J: JMP	IOTSK3
	2170	END	

USER SYMBOLS

AKREG 0331	AKREG1 0338	AKREG2 034C	AKREG4 0351	AKREG5 0376	AKREG7 0378	AKREG9 0385	AKREGA 038C
AR2 03A7	AR3 03B4	AR500 03BA	ARITLP 03A9	ARKENC 03A0	ARANC 03A2	ARTLP1 03BE	BAKREG 03CE
BOYSTA 0402	BOPE6 041A	BOACT 0440	BOATL 0430	BOCHECK 042C	BOCOUNT 0450	BOREG 0400	BEAULT 042X
BRREG 0650	BLKKB 0494	BRAXPD 0402E	BRCTYP 042F	BRGONE 04A3	BRGTHO 04A0	BTAREG 0600	BTAREG 067E
BTDPF8 0678	BTDP 0490	BTOTAL 0420	BTDDR 0400	CCE18 0253	CHAREN 0300	CHFAUL 05EC	CHGDIR 0411
CH788 0276	CHTDT 0408	CKBCHA 045A	CLP000 0163	CLR001 0170	CLR001 0205	CLKVR 015E	CLASH 04E1
CLPMN 040F	CHP00 0316	CHP01 0319	CHP02 032F	CHP03 0327	CHP04 0325	CHPLOT 0318	COMP0 0311
COMP0 030A	COMPEN 030E	CRBAT1 0208	DISALP 0281	DISCHA 021B	DISFRG 04FC	DISKR 048E	DISPAL 02E1
DMPTU 040E	DMPTIN 0401	DMPTIN 0400	DMPT0 0405	DONE33 0405	DONE 0794	DONE11 045D	DONE3 0408
DONE31 0406	DONE32 040F	DONE33 040E	DONE34 0408	DONE35 0408	DONE36 0402	DONE37 040C	DONE38 040D
DONE39 0408	DONE3D 0403	DONE3E 0405	DONE3F 040A	DONE5 0400	DONE51 040C	DONE52 0412	DONE53 0411
DP00 040A	DSALP1 020E	DSENP6 0304	DSFRG1 0105	DSK001 0401	DSK000 0400	DSK001 0400	DSK005 0408
EDAM1 040F	EDAM12 040F	EDAM2 041F	ENAFRG 0305	ENR10 0409	FAULT 0124	FAULT0 03E1	FAULT1 03E
FAULTC 030D	FDCK 0424	FDCK1 0421	FDCK1 042D	FDCK2 0428	FINBAT 0292	FINNO 0267	FINNO1 027
FINNO2 0255	FVCLY5 02E4	FVCHG 030C	HEAULT 05E4	INAKRI 0401	INIT 05E1	INIT1 05E7	INIT4 026
INITF 0774	INITP0 074C	INITP2 075C	INITPF 0761	INITPL 0737	INITPX 073E	INITZE 0400	IOR0K2 04C
IOR0P 0402	IOR0B1 0406	IOEND 0706	IOEND2 070D	IOERR 040E	IOERRM 070D	IOPTR 0437	IOR08 040
IOEPT 0401	IOTSK 0794	IOT0A 0416	IOTSK 0404	IOTSK0 0798	IOTSK1 0799	IOTSK3 040E	IOTSK5 07E
IOTSK6 040E	IOTSK8 07B1	IOTSKC 07B8	IOTSKD 07D1	IOTSK0 07C5	IOTSK5 0706	IOTSK2 0796	IOR08 03F
IOR0B1 03F5	IOR0B2 03FD	IOR0BJ 04D2	IOR0BX 0404	ITSK1J 0400	ITSK3J 07F0	ITSK01 07DF	KBTLG 01E
VP1G2 01C3	KBTG 04 01C1	KEY000 0212	KEY001 0216	KEY002 0218	KEY00 0204	KEY001 0290	KEY0F 02E
KEYSTA 0434	KEYTSK 0400	LSTD07 0428	LSTD09 042D	LSTD09 042F	MAIN 077F	MAIN2 078A	MAINJ2 02X
MAINR 0776	MAINR2 0779	METYP0 0417	MIMONE 04FF	MOV001 0175	MOV000 017E	MOV005 0170	MPST1 040
NE001 0100	NE002 010C	NE003 010E	NE004 0100	NE005 010C	NE001 0192	NVTD0R 0633	NOONEP FE1
N10TSK FFB	N10T FFF3	N10T FFF2	NODISP 040E	NORINT 01ED	NPETHC FFB	NPETTI FFB	NPHOL FFB
NPTREN FFB	NPTTSK FFB	NSTEP 0674	NSTEP0 067E	NSTEP1 067F	NSTEP2 068D	NSTEP3 068F	NSTEP5 068
NSTEP6 0692	NSTEP7 0699	NSTEP8 069D	NSTEP9 06A5	NSTEP0 06BE	NSTEP1 0675	NSTEP1 067E	NSWTSK EB

DRESHT 080F	PGRMOV 030E	PFTSX 0888	PHOCL 0882	POAC01 0288	POB006 029C	POBAT1 029D	POBATL 0295
POSDON 0808	PRCTR 0836	PRINT 0517	PRINT1 0587	PRINT3 0534	PRINT7 05C9	PRINT8 05C8	PRINT9 05CF
PROG1 0828	PROG2 0818	PROG2F 081F	PRTFRA 0554	PRTF3 0556	PRTF4 0564	PRTF5 0558	PRTF6 0568
PRTF7 0559	PRTF8 055E	PRTF9 055E	PRTFX 0568	PRTFY 0576	PRTSTA 0835	PTAEX 0648	PTITSX 0826
PTK208 080D	PTK201 0803	PTK202 0804	PTK204 0823	PTK206 0806	PTK211 0592	PTSK2 0538	PTSK05 0545
PTSK1 0548	PTSK2 0574	PTSK25 0825	PTSK26 0828	PTSK26 0588	PTSK25 0583	PTSK21 058C	PTSK2P 0689
PTSK32 0598	PTSK33 05A8	PTSK38 05A2	PTSK39 05A8	PTSK36 05B1	PTSK3X 059E	PTSK41 05F8	PTSK42 05FE
PTSK4E 05D1	PTSK4F 05DA	PTSK4H 05D6	PTSK4P 05F9	PTSK4T 05EB	PTSK51 061F	PTSK55 062D	PTSK57 062A
REFOPS 081F	REFDE6 081E	REFD0C 0145	REFD1E 015E	REFD01 0489	REFD21 0141	REFD31 013F	REFD8E 013C
REGDMS 0158	REG87 0828	REG89 082C	REG8J 0792	REGDSP 0434	REGED 03CC	REGSX 082E	RGD003 045D
RGD005 0465	RGD008 046F	RGD00A 043D	RGD00D 044F	RGD00E 044D	RGDSP1 0457	RGDSP3 0455	RGDSP4 0475
RGDSP 0479	SARAC 0829	SERIPH 0852	SHD 01D1	SHD 01DE	SHD1 01E2	SHD2 01F5	SHD3 01FA
SOLEN 0848	SOLFF 084F	STEPIN 0838	STEPTK 083C	STINTV 0884	STPDON 0828	STPM1 0888	STPTK2 083E
STPTRY 0848	SWRIBA 087F	SWTSK 0818	TIM000 0854	TIMONT 08FB	TIMFIN 08EA	TIMING 083E	TIMINH 0848
TIMINE 0859	TIMINT 0867	TIMFET 08F1	TIMINT1 0818	TIMINT2 0839	TIMINT3 082E	TDEESS 01EB	TDEFST 01EB
TOPTI 01AF	TOPTIC 01AC	VER 0839	VER1 064A	VER2 0653	VER3 066A	VER5 065C	VER8 066F
VERLP 0645	WATT 06FB	WATT1 072F	WPD000 011D	WPD001 0126	WRITB1 0112	WRITB2 0133	WRITB5 0139
WRITB4 080F	WPTRM1 0108	WPTRE 0108	WTDSP 08F1	WTDSP1 08F8	XCHPT 056C	ZERO9 0893	

ASSEMBLY COMPLETE. NO ERRORS



INITIZ 1831									
IOB2 1418 14121									
IOPE 1531									
IOPE2 1482 14211									
IOEN 21441 2167									
IOEN2 1485 21481									
IOEE 1204 1413 14171 2158									
IOERR 2132 21581									
IOPTR 681 1482 2114 2165									
IOPE 1541 2137 2162									
IOPTF 1521									
IOPTS 2878 21881									
IOSTE 671									
IOYSK 881 89 363									
IOYSK2 1418 21111 2161									
IOYS1 21131 2124 2125 2134									
IOYS3 13561 2169									
IOYS5 1218 21651									
IOYS6 1421 14251									
IOYS2 2127 21291									
IOYS1 2132 21351									
IOYS0 2129 21581									
IOYS4 2136 21411									
IOYS5 21221									
IOYS7 2184 2145 2168									
IOYS 12821 1428									
IOPE1 1283 12851									
IOPE2 1287 12181									
IOPE3 1482 14191									
IOPE4 1551									
IOYS1 1391 14181									
IOYS3 2153 21691									
IOYS01 2157 21591									
KBTL6 6391 1837									
KBTL3 644 6461 869									
KBTL4 642 6451									
KEY82 732 7481									
KEY81 748 7421									
KEY82 742 7441									
KEY8 7311 2882									
KEY80J 2879 28831									
KEY8F 645 788 7921 885 911 912 914 1838									
KEY8F 511 792 1755 1889 1995 2857									
KEYTS 821 96 99 371									
LSTC87 581									
LSTC29 511 781									
LSTCS 521									
M2IN 28741 2881									
M2IN2 2877 28881									
M2INJ2 797 7991 885 893									
M2INF 886 1335 1569 1686 28531 2149									
M2IN2 798 1558 1672 28661									
M2TYPE 551 648 1845 1989 2853									
M2INONE 381 247 1145 1968									
M2V21 5781 821 897 899 1829 1447									
M2V22 5841 887 1437 1441 1445 1722 1768									
M2V23 5891 874 1456 1739									
M2ST1 311 798									
M2EB1 579 5861									
M2EB2 591 5941									
M2EB3 581 5951									
M2EB4 599 6821									
M2EB5 681 6841									

0019515

NEST	500	500	604						
NTDIF	884	1714	1766						
NDONE	144	1811	1824	1853					
NDITS	29	90	99	2148					
NDIOT	99								
NDIOP	90	799							
NDIOP	212	388							
NDIOT	187	696							
NDIOT	97								
NDIOT	98	626							
NDIOT	94	97	1663						
NDIOT	96	97	96	1685					
NDIOT	87	90							
NDIOT	1641	1839							
NDIOT	1846	1874							
NDIOT	1847	1850	1876						
NDIOT	1856	1865							
NDIOT	1854	1862							
NDIOT	1852	1862							
NDIOT	1864	1867							
NDIOT	1868	1871							
NDIOT	1841	1878							
NDIOT	1843	1875							
NDIOT	1851	1857							
NDIOT	1844	2040							
NDIOT	141	1864							
NDIOT	85	1334							
NDIOT	390	1231							
NDIOT	1145	2120							
NDIOT	92	702							
NDIOT	93	94	357						
NDIOT	850	570							
NDIOT	856	858							
NDIOT	855	857	859						
NDIOT	854	855							
NDIOT	143	144	1821						
NDIOT	59	194	622	1959	2000				
NDIOT	1495	2002							
NDIOT	1507	1660							
NDIOT	1511	1515							
NDIOT	1672								
NDIOT	1671	1673							
NDIOT	1672	1675							
NDIOT	26	667	702						
NDIOT	23	25	287	1147					
NDIOT	25	473							
NDIOT	1541	1542	1551	1589	1593	1594	1612	1613	1643
NDIOT	1552	1559							
NDIOT	1559	1561							
NDIOT	1554	1560							
NDIOT	1557	1563							
NDIOT	1555	1562							
NDIOT	1556	1925							
NDIOT	1558	1564	1696	1716					
NDIOT	1565								
NDIOT	1583	1568							
NDIOT	62	650	795	1500	1555	1922	2025		
NDIOT	95	96	1235						
NDIOT	86	87	282						
NDIOT	1909	1912							
NDIOT	1911	1917							
NDIOT	1913	1918							
NDIOT	1891	1893							

BAD ORIGINAL



0019515

PTK28	1647	1748	1922
PTK2L1	1614		
PTSK8	1523		
PTSK85	1533		
PTSK1	1516	1538	
PTSK2	1517	1579	
PTSK25	1894	1897	
PTSK26	1901	1921	
PTSK26	1584	1590	
PTSK29	1592	1648	
PTSK2L	1610	1615	
PTSK2P	1618	1886	
PTSK32	1586	1622	
PTSK33	1623	1625	
PTSK38	1591	1635	
PTSK39	1637	1638	1641
PTSK3A	1617	1644	
PTSK3Y	1496	1624	
PTSK41	1658	1713	
PTSK42	1781	1717	
PTSK4E	1662	1682	
PTSK4F	1684	1687	
PTSK4H	1685	1690	
PTSK4P	1786	1714	
PTSK4T	1669	1695	
PTSK51	1675	1745	
PTSK55	1749	1758	1755
PTSK57	1747	1753	
R2FORG	43	435	1178
R7FORG	44	1284	1419 1584 2818 2144
REF001	518		
REF051	525	537	
REF081	1345	1347	
REF091	513	516	
RED0M1	417	514	598 1884 1121 1894
RED0M2	512	954	1977 2163
RED0M5	529	643	867 994 1119
REG087	45	732	1238 1956
REG09	46	268	552 735 1992
REG0S1	2876	2884	
REG0SP	1283	2884	
REG0ED	981	1817	1156 1238
REG0SU	48	362	1298 1526 1682
RGD003	1388	1310	
RGD005	1321	1327	
RGD009	1323	1329	1338
RGD00A	1286	1288	
RGD00O	1299	1382	
RGD00F	1298	1339	
RGD0P1	1385	1341	
RGD0P3	1299	1383	
RGD0P4	1293	1334	
RGD0PP	1296	1336	
SAVACC	69	399	1955
SEPIPH	73	1386	
SND	661	1538	1642 2838
SND8	671		
SND1	678	674	
SND2	669	677	
SND3	677	681	
SOLEH	44	15	286 472 516 529 1666 786 787 1146 1948
SOUFF	151	1475	
STEPIN	72	1467	1893 2844 2858

BAD ORIGINAL

0019515

STEPT	751	1456							
STINTV	1454	621							
STPDON	1424	144	1812						
STPM1	1394	141	1855						
STPTK2	784	1686							
STPTRY	1474	1587							
SWFBR	224	489							
SWTSK	844	85	1156	2817					
TIM800	265	2674							
TIMCH	374	192	1961						
TIMFIN	261	269	282	3984					
TIMINE	226	223	2734						
TIMIN1	222	2464							
TIMIN5	228	241	2734						
TIMINT	1994								
TIMPET	362	394	397	3994					
TMINT1	196	195	2044						
TMINT2	215	2274							
TMINT7	2114	288							
TOFFSS	649	781	7654	1191	1565	1674			
TOFFST	7874								
TOPTI	6214	661	1782						
TOPTIC	6224	1646	1786	1714					
VER	1579	17814							
VER1	1794	17964							
VER2	1797	18834							
VER3	1818	18204							
VER5	1886	18104							
VER6	1884	18244							
VERLP	17924	1882							
WPT	19714	1972							
WRT1	28134	2815							
WRS800	476	4784							
WRS801	483	4864							
WRTB1	465	4724							
WRTB2	493	4954							
WRTB5	495	4984							
WRTB6	214	488							
WRTB11	4644	558	566	582	688	997	1885	1811	1134
WRTB2	4654	594	682	685	1198	1289			
WRTB5F	4154	425	1318	1372					
WRTB5P1	422	4244							
XCHPT	14654	1515	1554	1616	1981	1918			
ZER09C	344	1322							

CROSS REFERENCE COMPLETE

BAD ORIGINAL



1/23

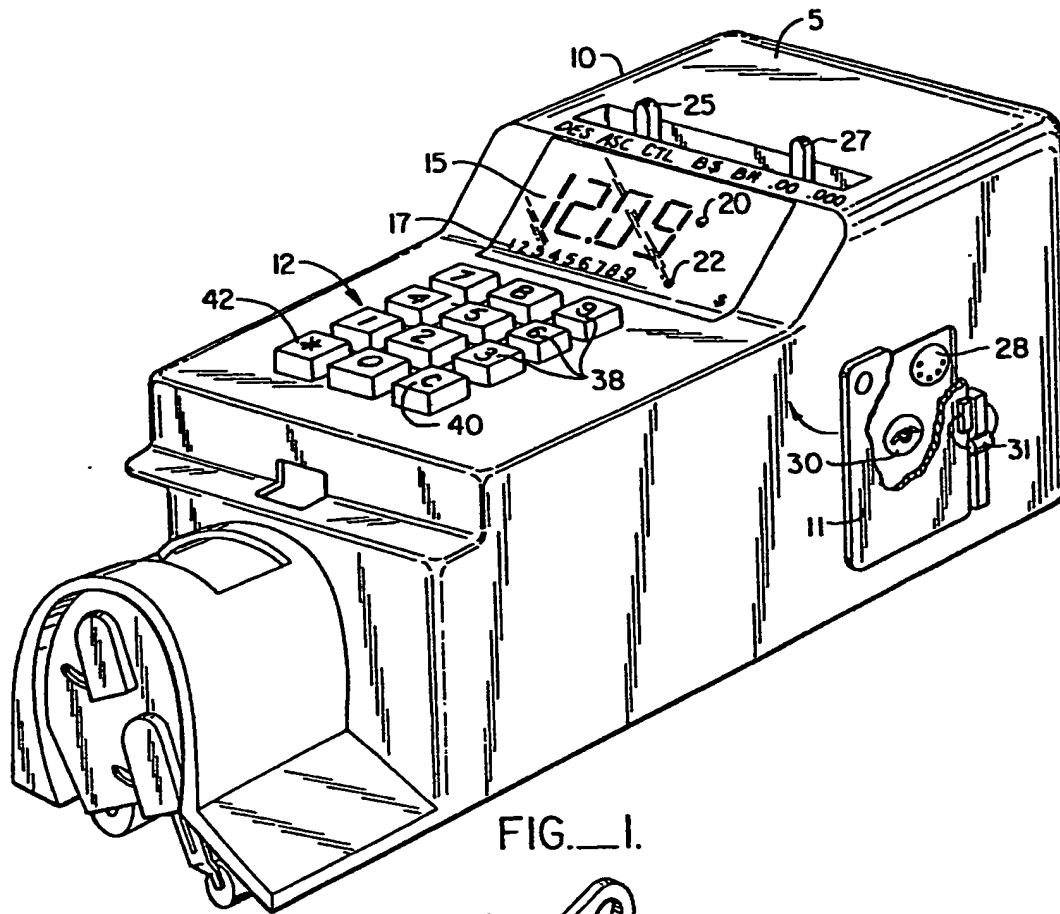


FIG. 1.

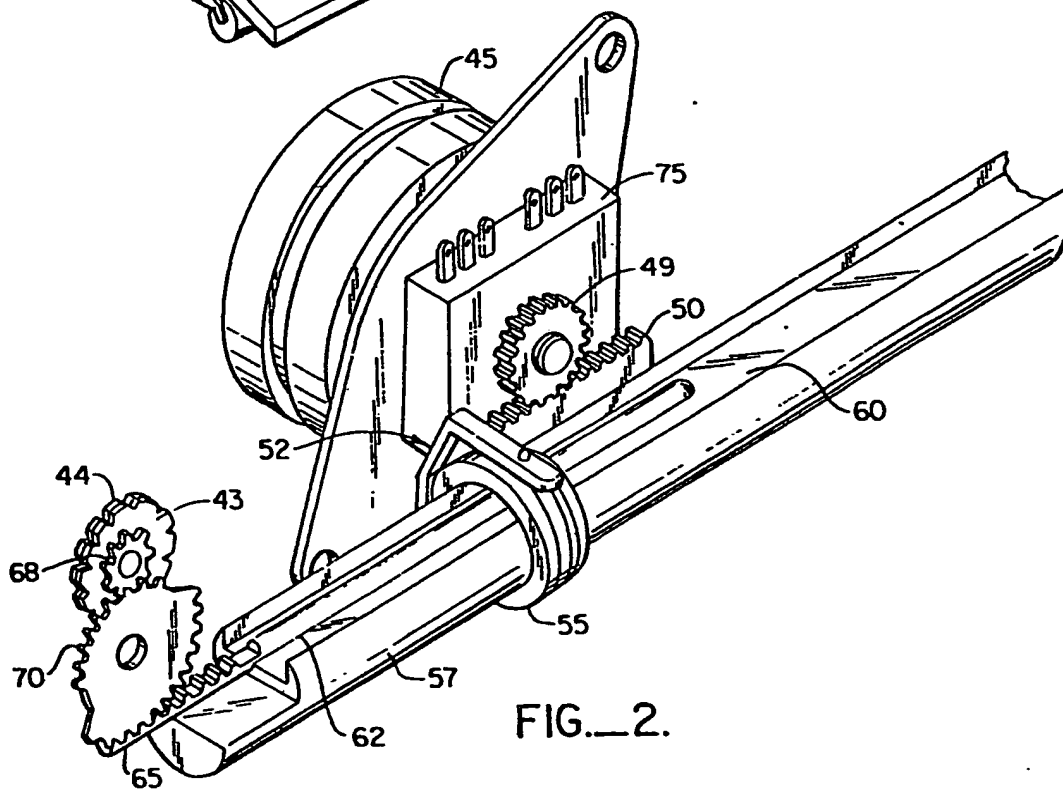


FIG. 2.

2/23

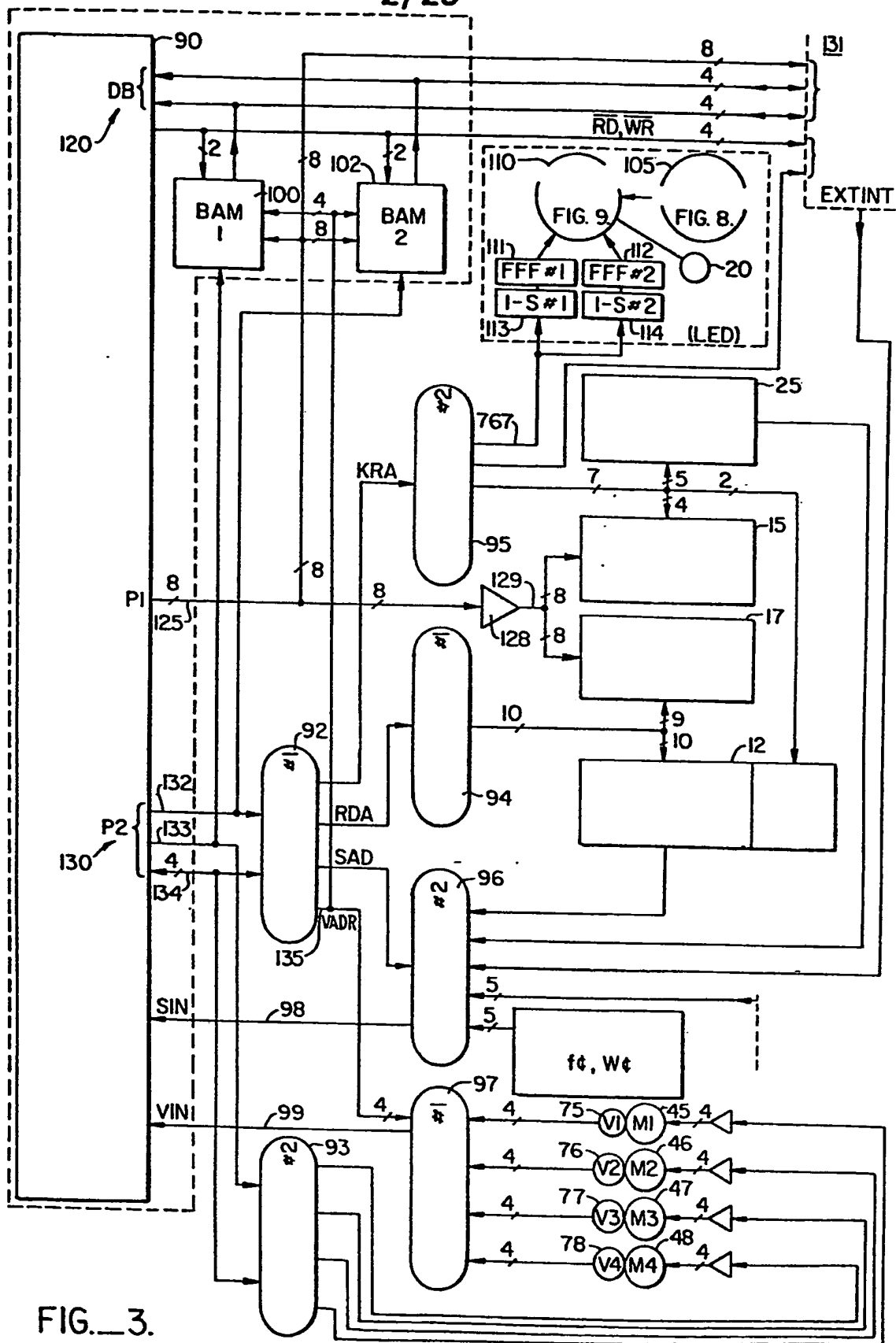


FIG. 3.

3/23

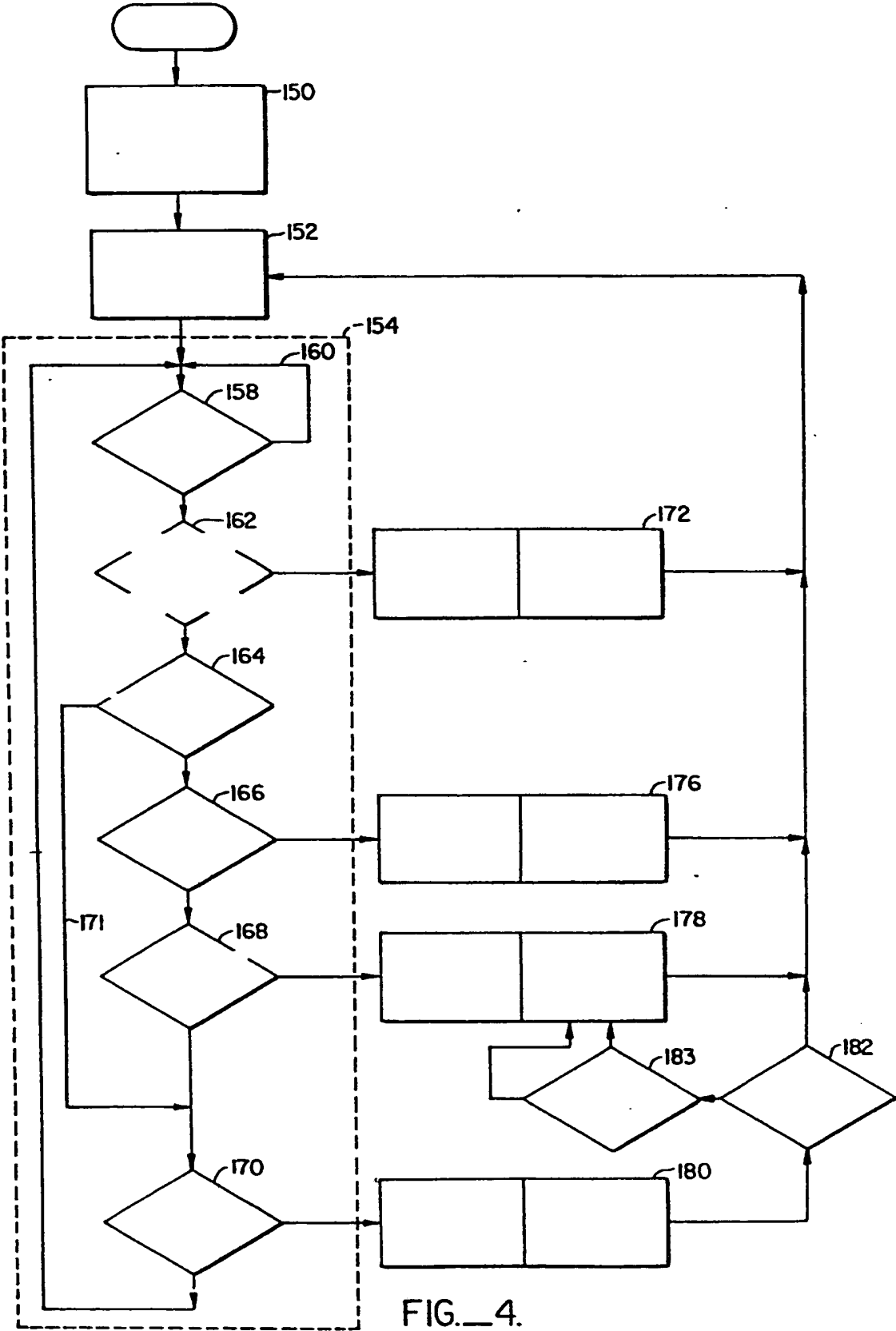
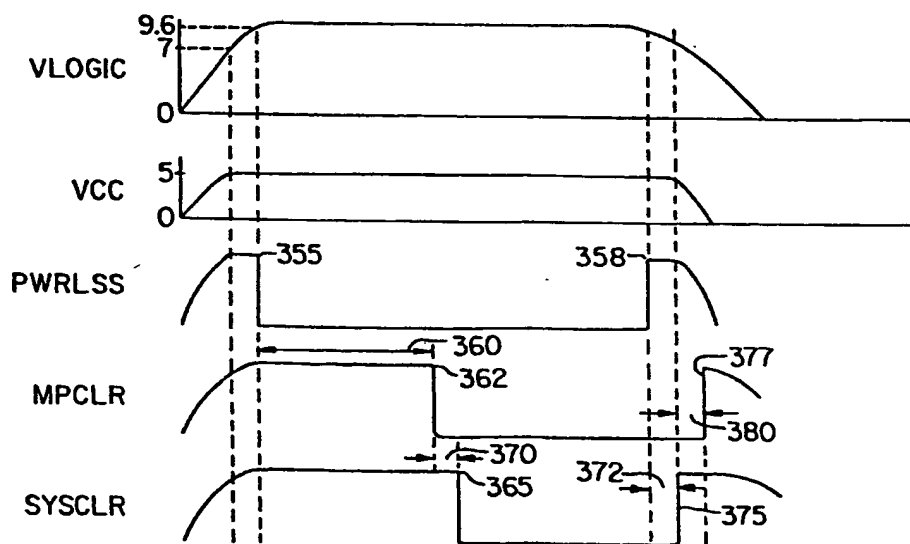
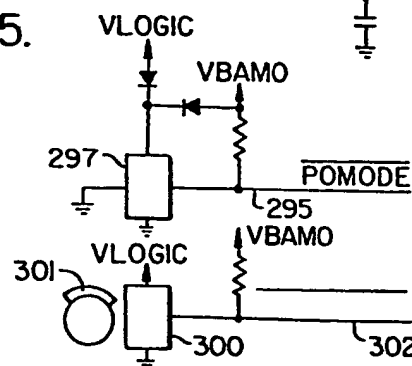
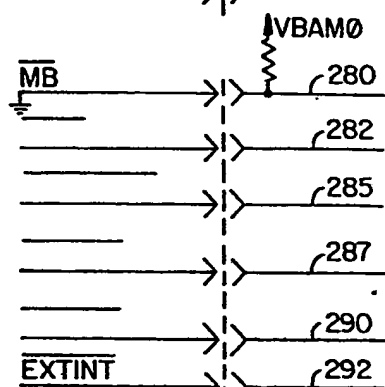
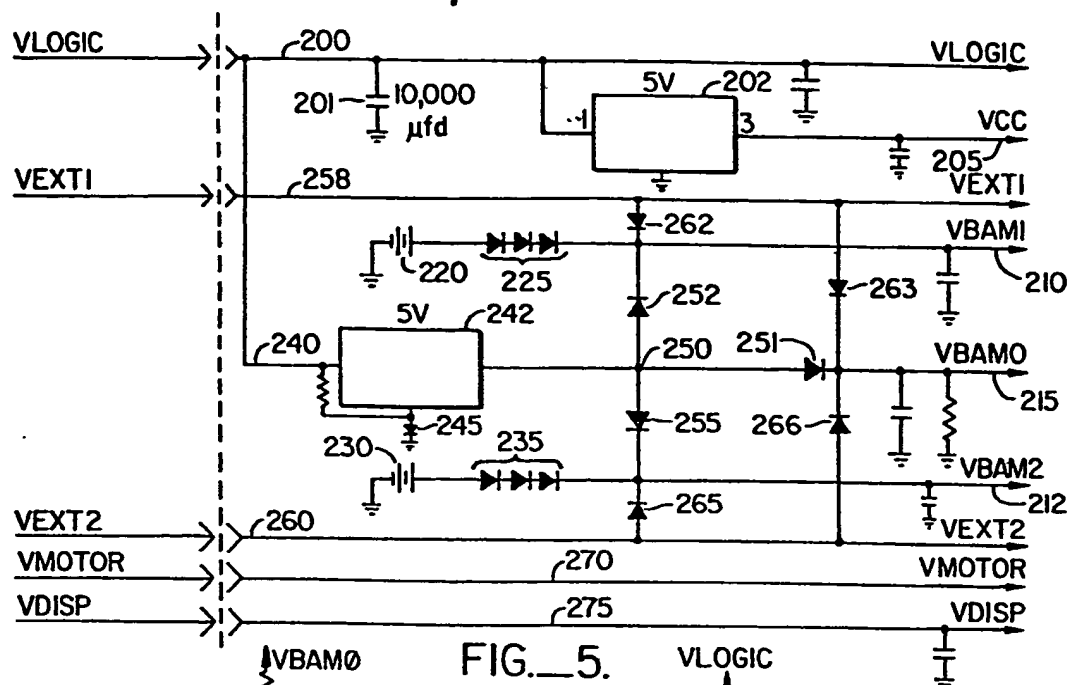


FIG. 4.



5/23

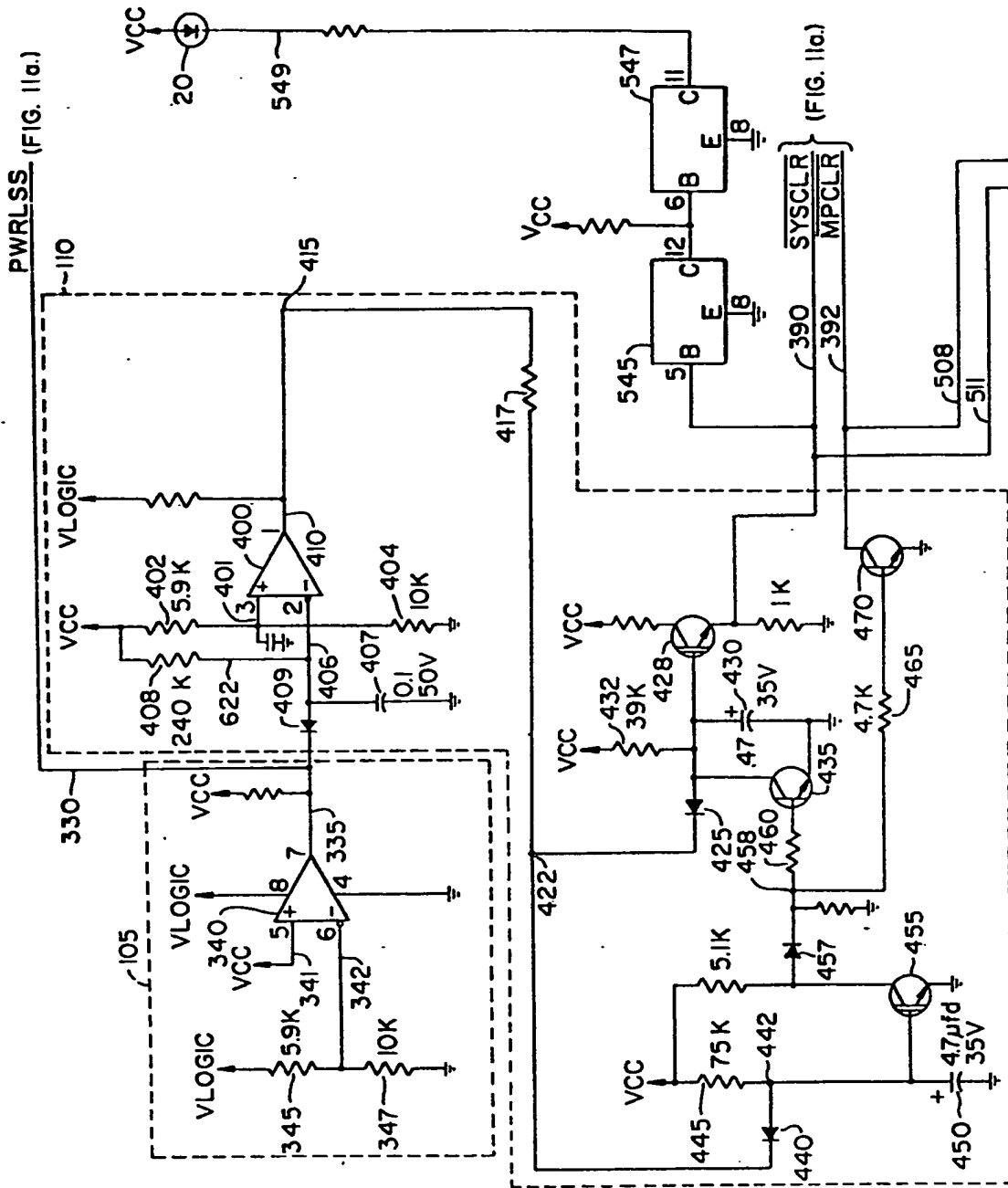


FIG.—8.

6/23

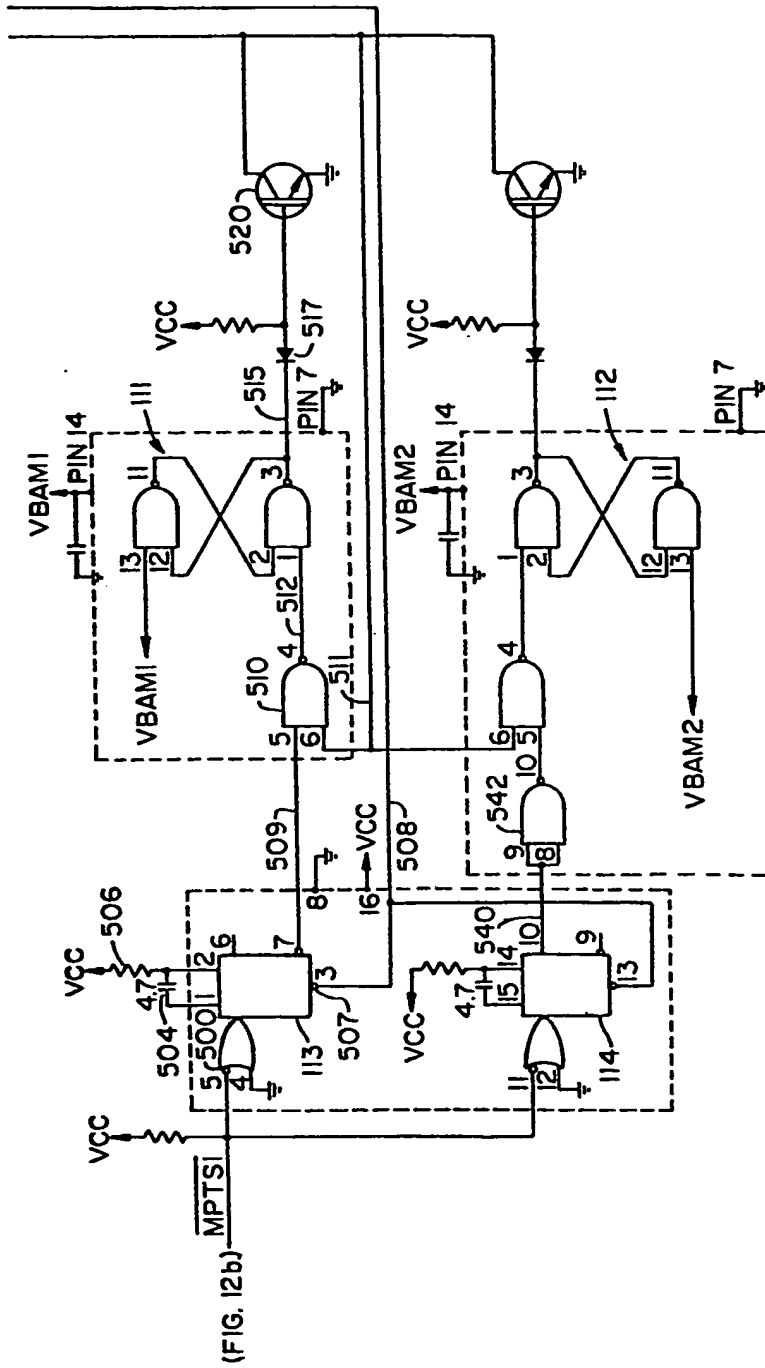
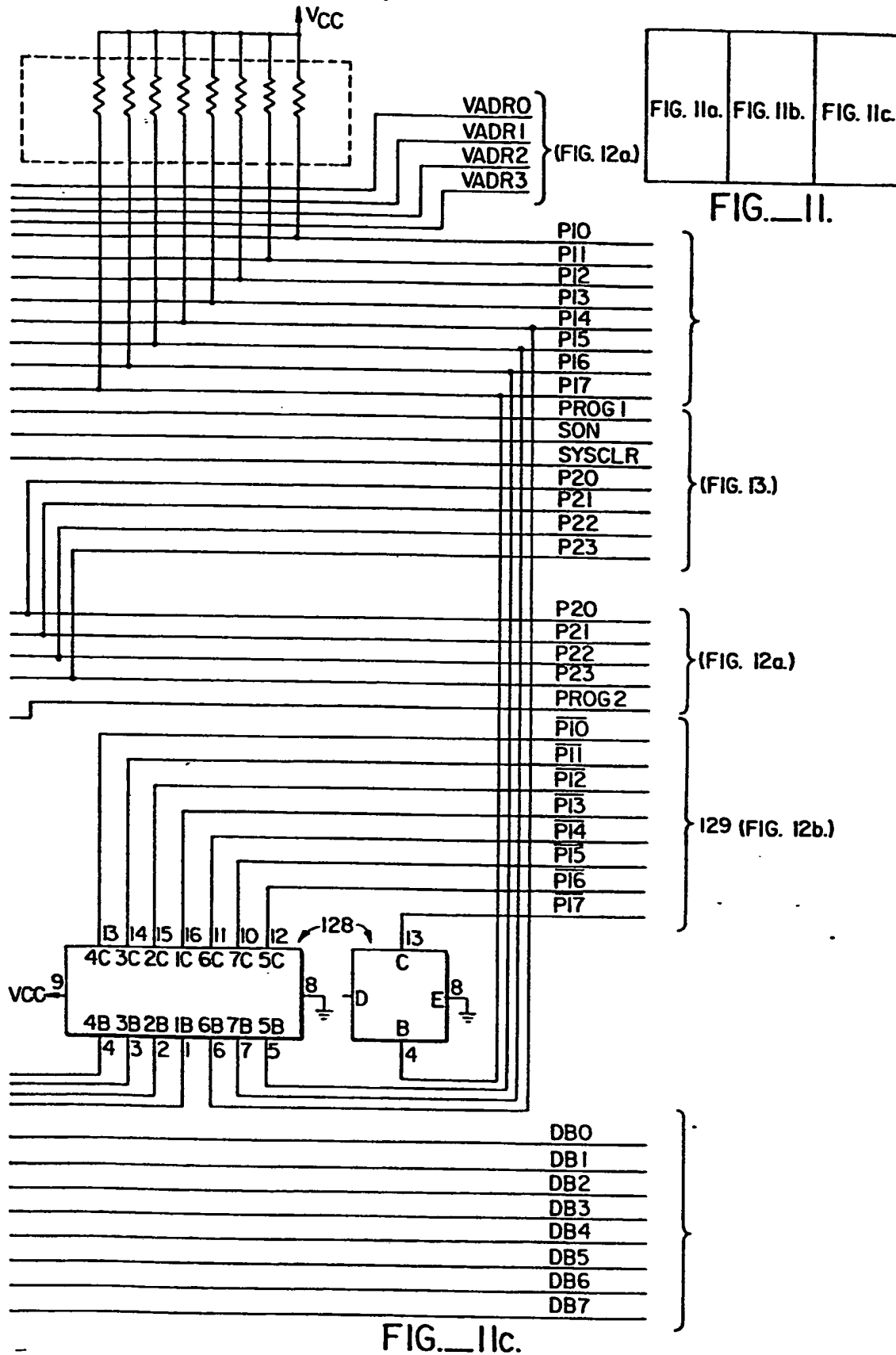


FIG. 9.

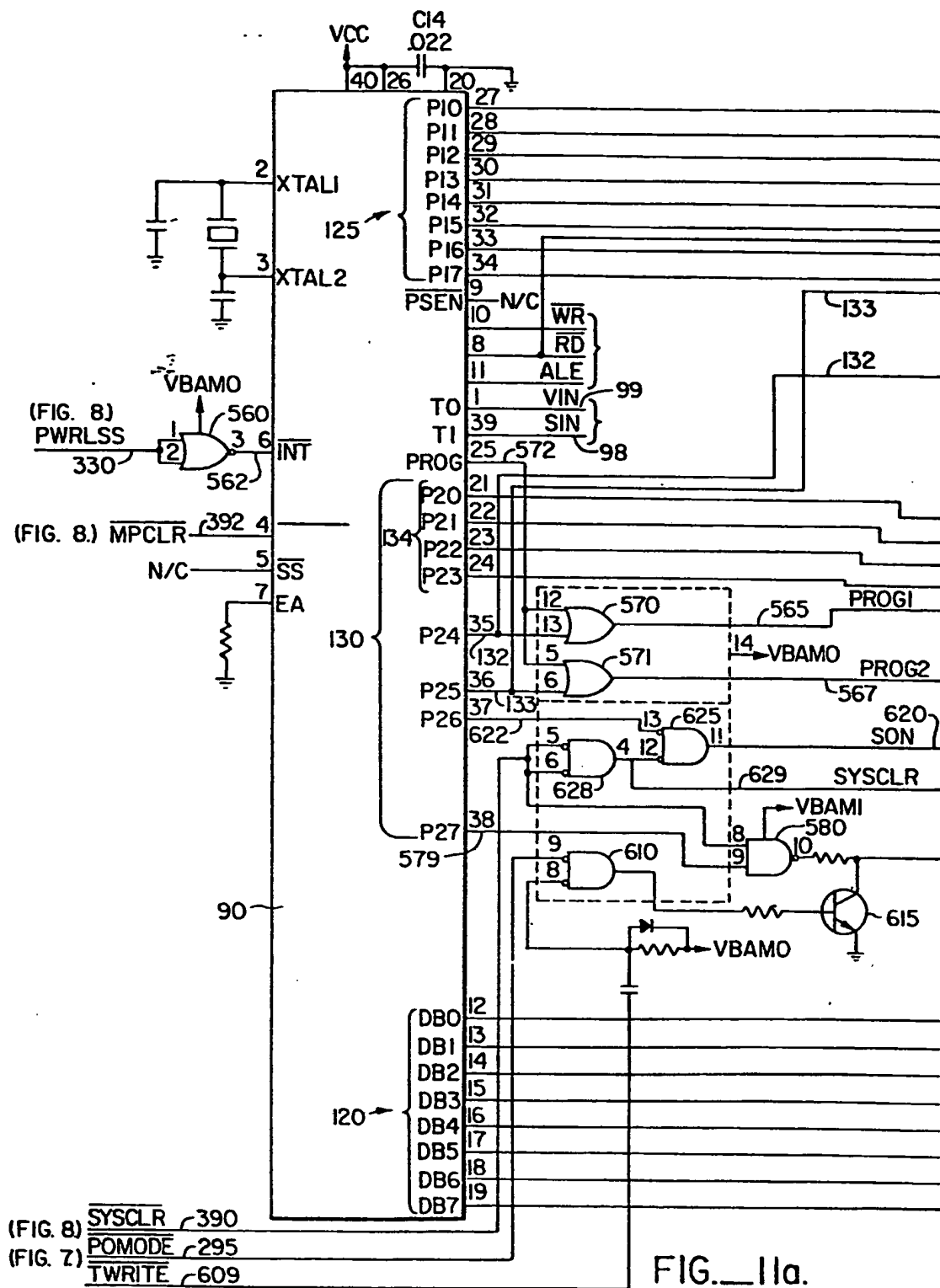
FIG. 8.
FIG. 9.



7/23



8/23





10/23

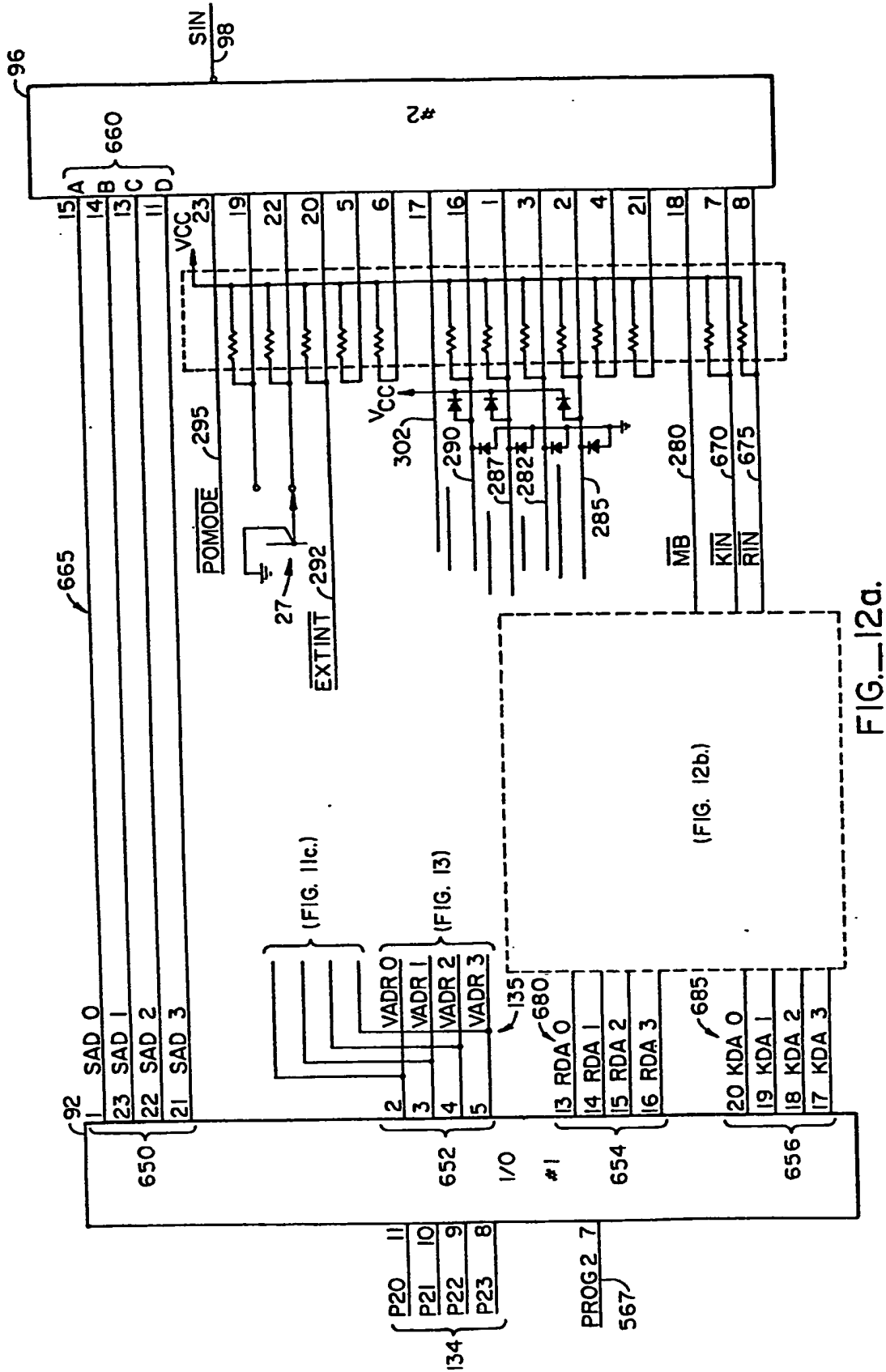
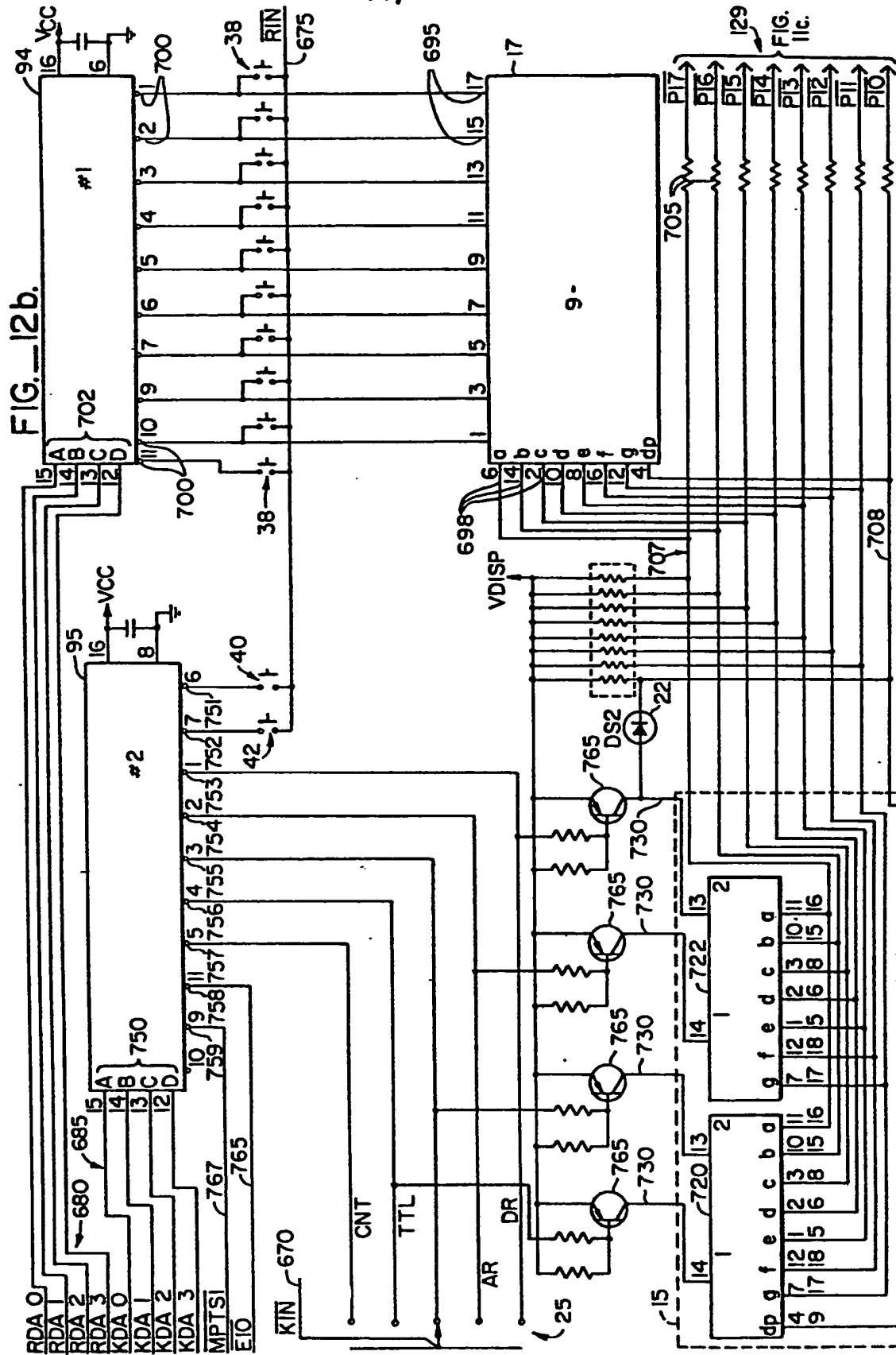


FIG. 12a.



12/23

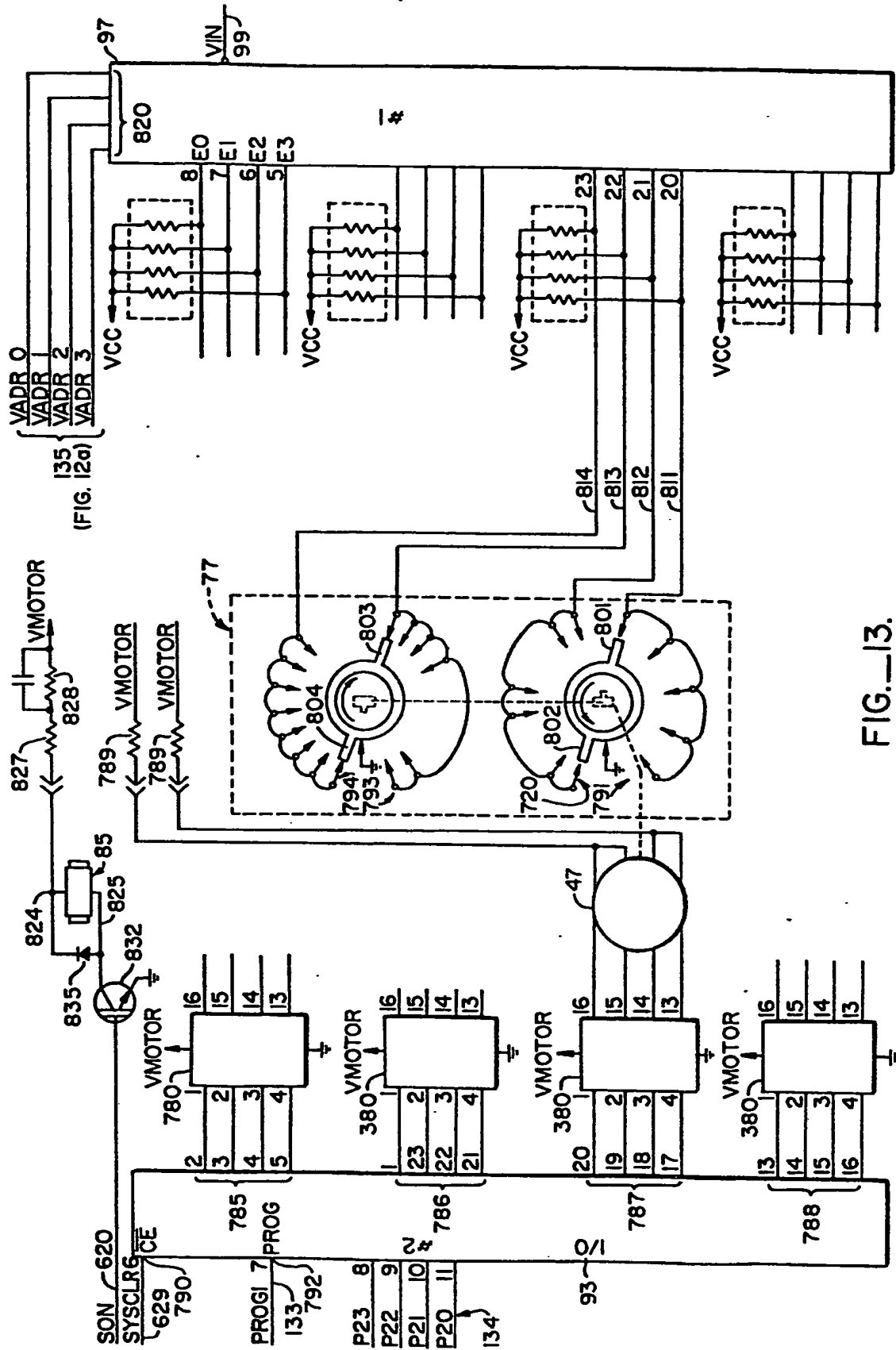


FIG. 13.

13/23

7	0	7	4	3	0
0 R0	—	20 LSD			
1 R1	—	21			
2 R2 BAKSTA	—	22			
3 R3	—	23			
4 R4	—	24	9		
5 R5	—	25			
6 R6	—	26			
7 R7	—	27			
8	—	28 MSD			
9	—	29 SAVACC			
A	—	2A	REG07		
B	—	2B			
C	—	2C	REG89		
D	—	2D			
E (6)	—	2E	REGSW		
F	—	2F			
10	—	30 LSD			
11	—	31	4		
12	—	32			
13	—	33 MSD			
14	—	34 KEYSTA			
15	—	35 PRSTA			
16	—	36 PRCTR			
17	—	37 IOPTR (I/O)			
18 R0'	—	38 STEPIN (0)			
19 R1'	—	39 STEPIN (1)			
1A R2'	—	3A STEPIN (2)			
1B R3'	—	3B STEPIN (3)			
1C R4'	—	3C STEPTK (0)			
1D R5'	—	3D STEPTK (1)			
1E R6'	—	3E STEPTK (2)			
1F R7' R7FORG	—	3F STEPTK (3)			

FIG. 14.

14/23

BIT	R7FORG (1F HEX)	REG07 (2A HEX)	REG89 (2C HEX)	REGSW (2E HEX)
0		"0"	"8"	DR
1	EXTINT	"1"	"9"	AR
2	.000	"2"		
3	MB	"3"		
4		"4"	POMODE	
5		"5"	.00	
6	"1"	"6"	FCENTS	
7	"1" REG07	"7"	POMODE /FCENTS .00	

FIG. 15.

15/23

BIT	BAKSTA 2 HEX	KEYSTA 34 HEX	PRTSTA 35 HEX	(38-3B HEX) (0-3)	STEPTK (0-3) (3C-3F HEX)
0					
1			(0-3)		
2		CC	"0" "1"	(BCD)	(BCD)
3					
4					"1" "
5		.00	(0-7)		"1"
6		FCENTS			"1"
7	PWRLSS	/FCENTS .00	"0" "1"		5 "0"/"1" "1"

FIG. 16.

16/23

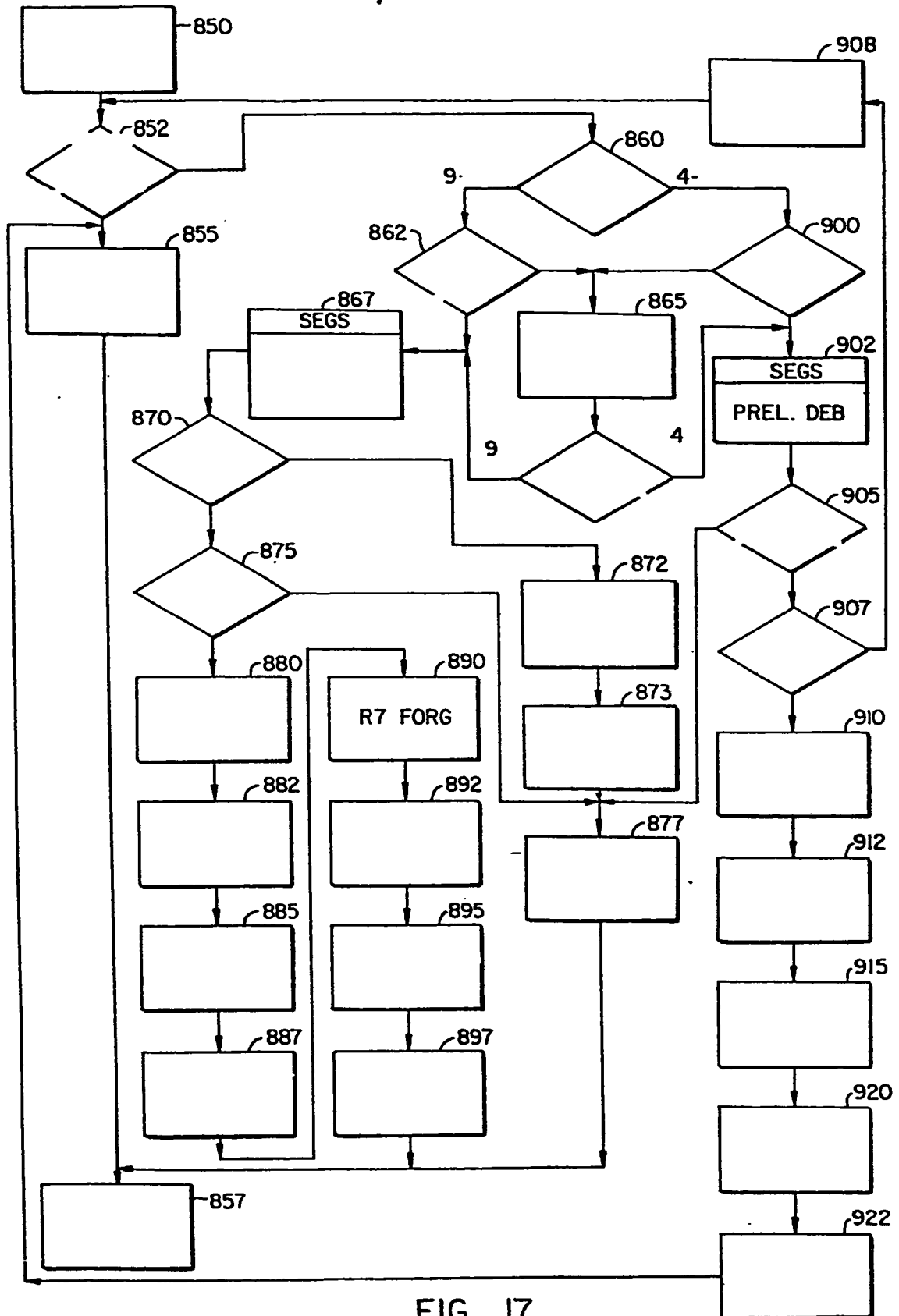


FIG. 17.

17/23

BAM1																BAM2																															
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F																
00	REG 1									*** 123							REG2									*** 456																					
DR 1									DR2																																						
ARI									AR2																																						
10	REG1																REG2																														
2																2																															
20	1																2																														
30	BS1																BS2																														
40	B#1																B#2																														
50	1																2																														
60	KB									*7																																					
REG																																															
TEMP DESC REG1																TEMP DESC REG2																															
TDR1																TDR2																															
70	TEMP ASC REG1																TEMP ASC REG2																														
TARI																TAR2																															
80	TEMP REG1																TEMP REG2																														
TEMP1																TEMP2																															
90	REGTH01																REGTH02																														
A0	REGONE1																REGONE2																														
0 1 2 3 4 5 6 7 8 9 A B C D E F																0 1 2 3 4 5 6 7 8 9 A B C D E F																															
*1 BAM1																*4 BAM2																															
*2 MAXPO BAM1																*5 MAXPO BAM2																															
*3 METYP BAM1																*6 METYP BAM2																															
																*7 KBCC																															

FIG. 18.

	TDR1	TDR2	TAR1	TAR2		
CHKTOT	DR	DR	AR	AR	AR+DR	
	DR-KB	DR	AR+KB	AR	(AR+KB) + (DR-KB)	
	DR	DR	AR	AR	AR+DR	
	DR±KB	DR	AR	AR	(DR±KB) + AR	(AR+DR) ± KB

FIG._19.

19/23

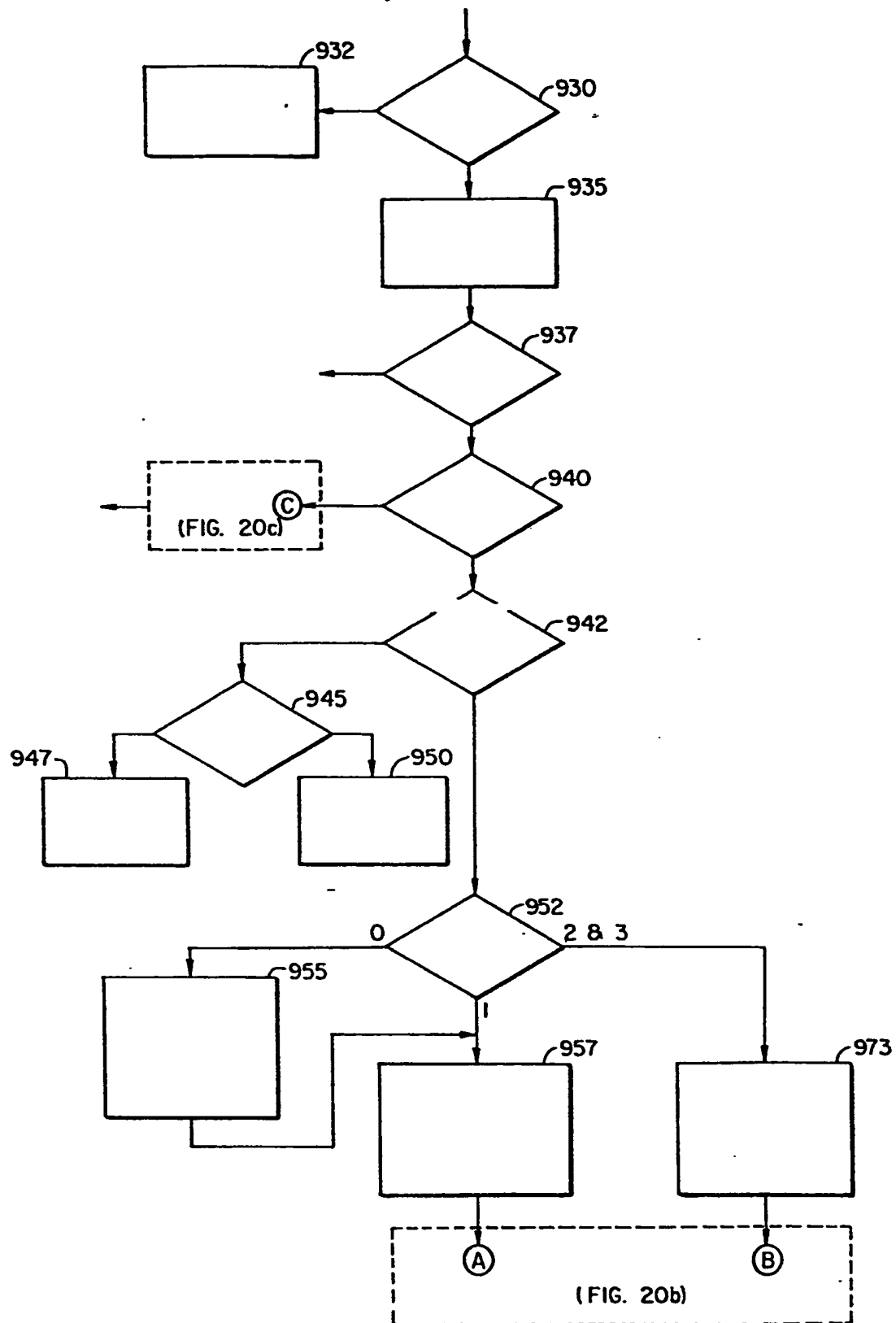


FIG. 20a.

20/23

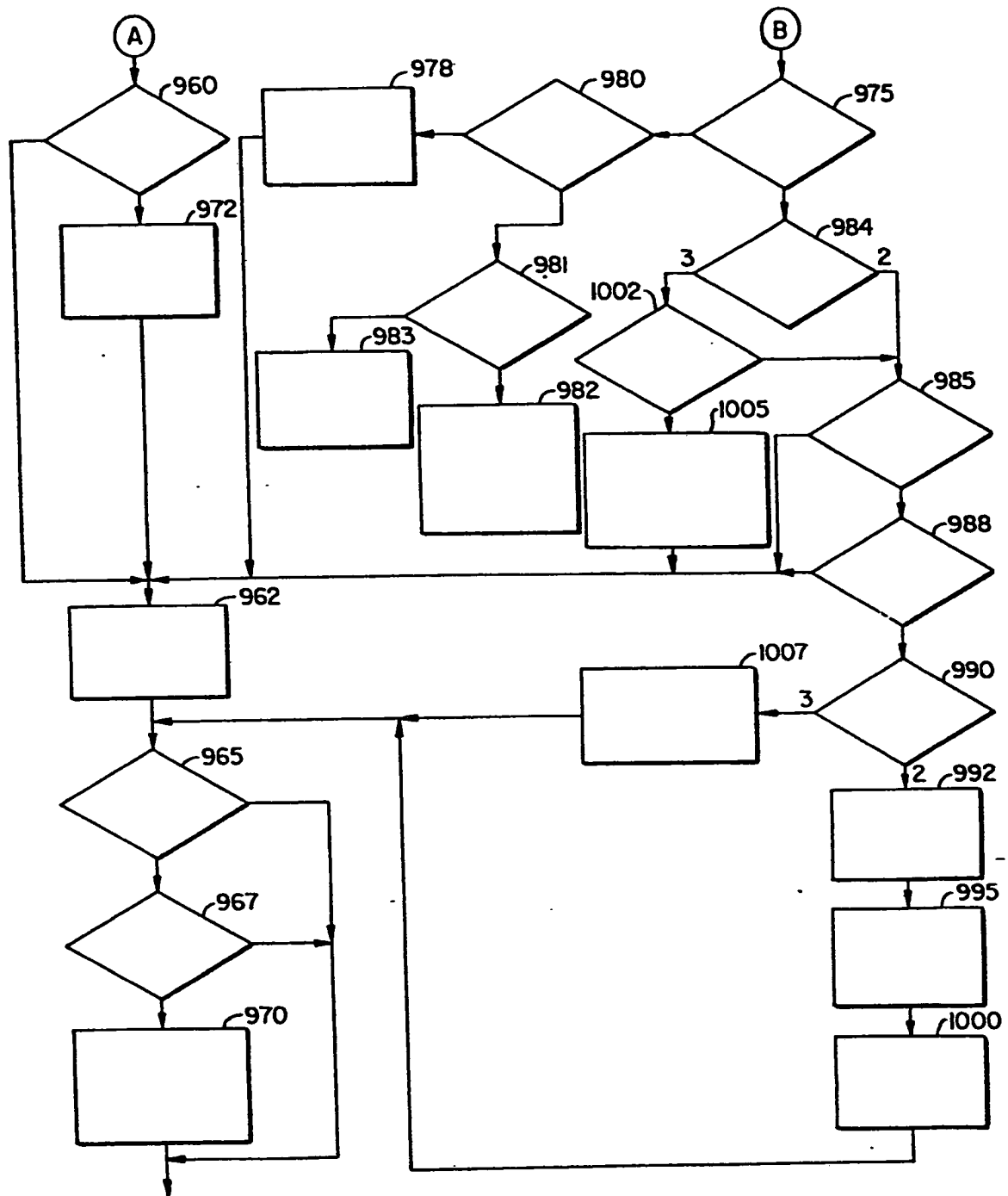


FIG. 20b.

21/23

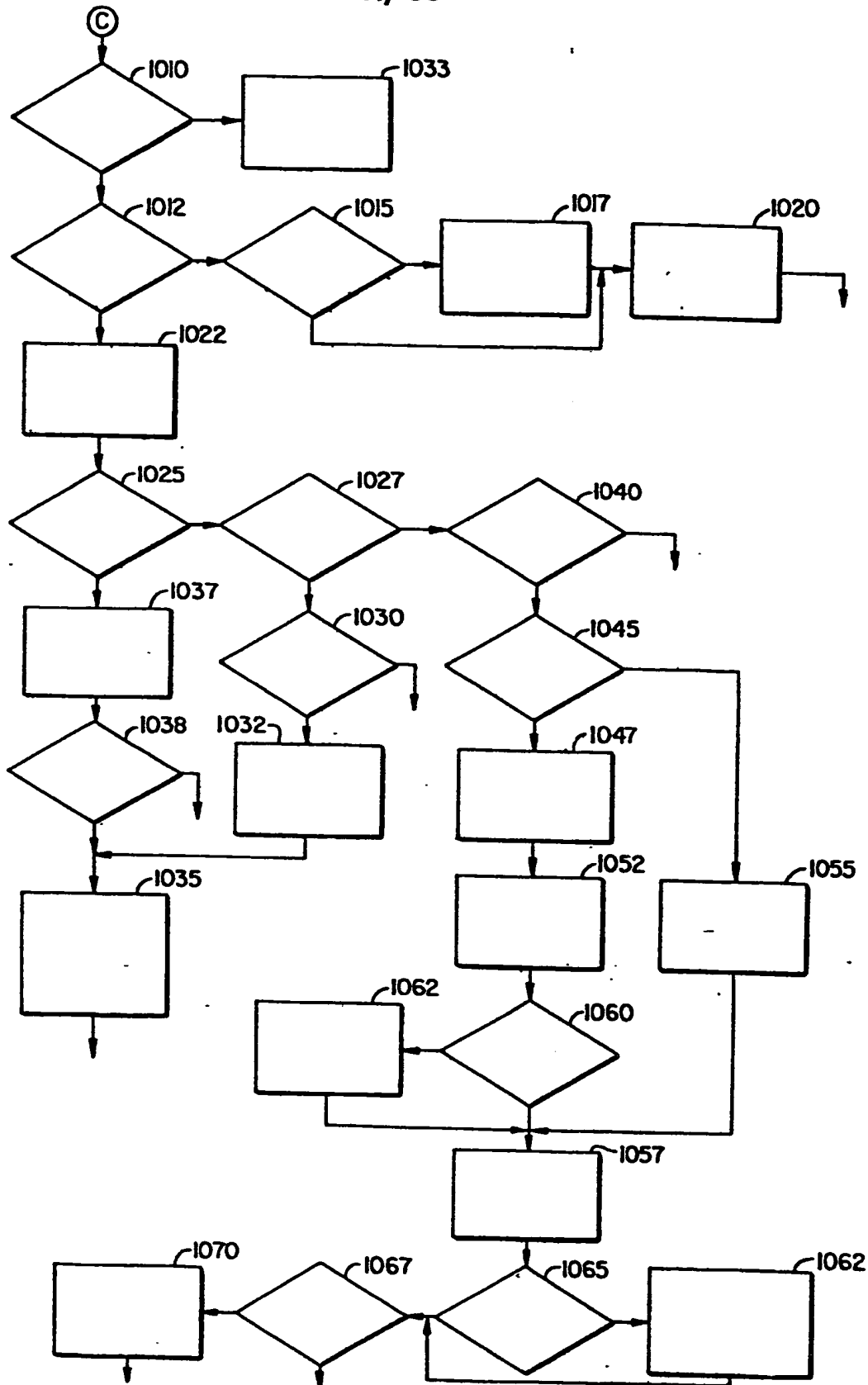


FIG. 20c.

22/23

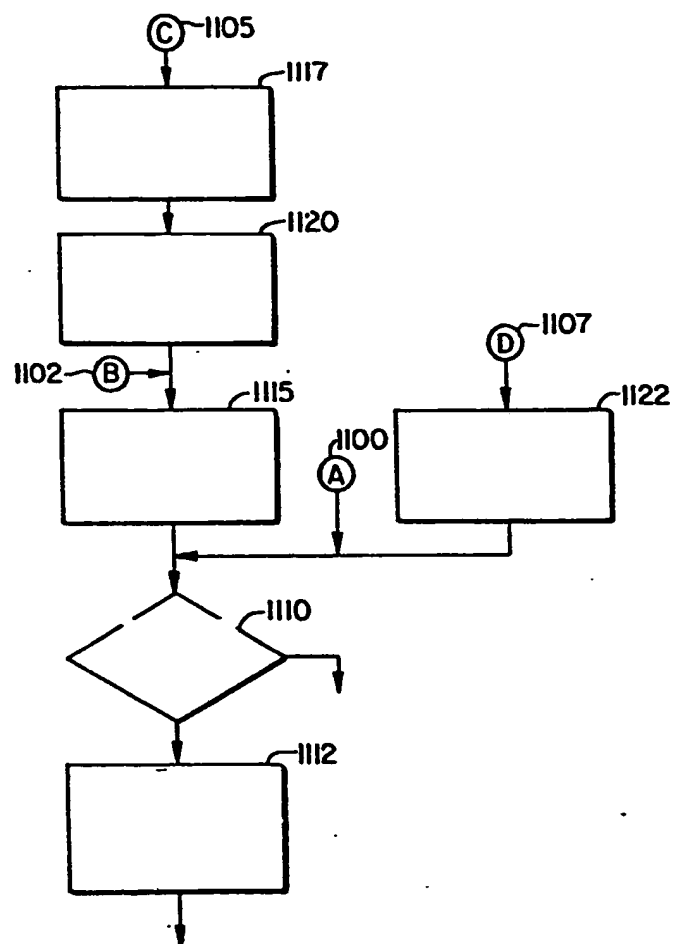


FIG. 21a.

23/23

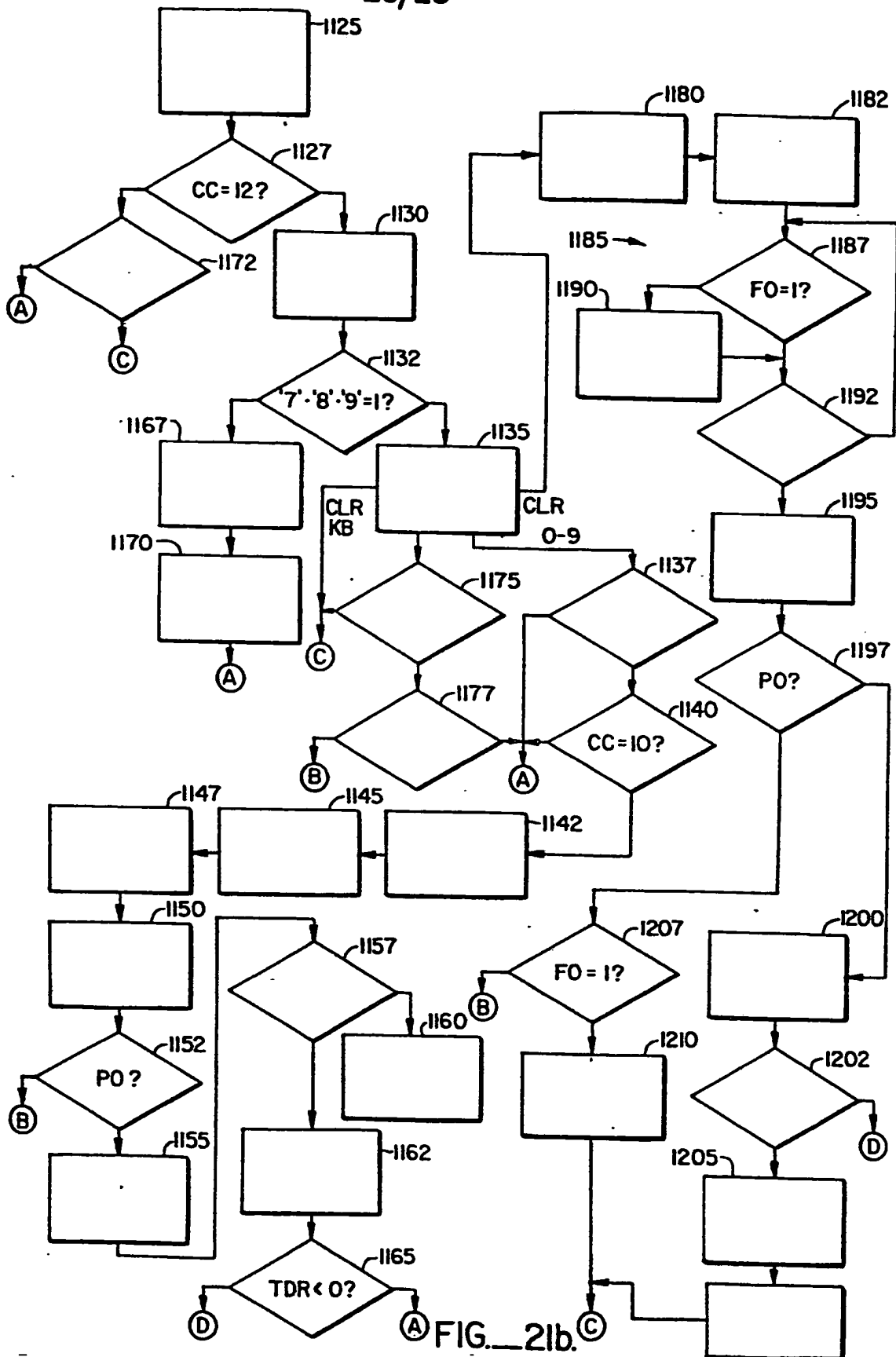


FIG. 21b.